Universidade do Minho
Escola de Engenharia

Rita de Abreu Loureiro

Internet das Coisas: Arquiteturas, plataformas e estudo de casos

Dissertação de Mestrado
Mestrado em Engenharia Eletrónica Industrial e de Computadores

Trabalho efetuado sob a orientação do
Professor Sérgio Lopes

Dezembro de 2019

DECLARAÇÃO DE INTEGRIDADE

Declaro ter atuado com integridade na elaboração do presente trabalho académico e confirmo que não recorri à prática de plágio nem a qualquer forma de utilização indevida ou falsificação de informações ou resultados em nenhuma das etapas conducente à sua elaboração.

Mais declaro que conheço e que respeitei o Código de Conduta Ética da Universidade do Minho.

AGRADECIMENTOS

À minha mãe, que sempre me apoiou e ajudou neste capítulo da minha vida que se encerra com a apresentação desta dissertação.

Aos meus avós, pelo amor e orgulho que têm na sua única neta.

Ao Miguel, pela carinho e dedicação a uma menina que lhe é muito querida.

Ao professor Sérgio Lopes, pela total disponibilidade e atenção que dedicou na ajuda à realização desta dissertação.

À Fujitsu, nomeadamente ao Filipe Guerreiro e ao André Figueiredo, por todas as horas que disponibilizaram para me ajudar e pelos conhecimentos que fomos trocando.

E por fim, à F3M que há dois anos me desafiou a iniciar um projeto inovador dentro da empresa, e do qual tenho imenso orgulho.

RESUMO

A base do ecossistema da Internet das Coisas vai muito para além de dispositivos, cidades, objetos e infraestruturas interligadas e em constante comunicação. A Internet das Coisas integrará e utilizará tais elementos e seus dados no progresso de novos serviços e produtos, que podem alterar e ter impacto na vida quotidiana.

Dado tratar-se de uma área recente em constante expansão e mutação, as tecnologias, conceitos-chave e desenvolvimentos são sistematicamente revistos. Portanto nesta dissertação um dos focos principais é inteirar do estado da arte tendo em vista que, este é um conceito que suscita o interesse de empresas mundialmente influentes que se unem para padronizar e normalizar a Internet das Coisas.

Numa primeira instância são discutidos o conceito e a aplicabilidade da Internet das Coisas, abordando posteriormente os seus princípios. A investigação de arquiteturas referência permite adquirir um conjunto de informações sobre as estratégias a adotar em relação à Internet das Coisas, compondo assim um mapa relacional de plataformas de middleware para IoT, protocolos e plataformas IoT. Os desafios inerentes a este paradigma são explorados levando a uma perspetiva de discussão, que se transponde para as direções futuras tanto a nível de hardware como de software.

De modo a permitir uma avaliação prática das características referentes à Internet das Coisas, foram selecionadas soluções para experimentação, desenvolvendo posteriormente um caso de estudo com os respetivos resultados.

Palavras-Chave: Internet das Coisas; Desafios; Tecnologias IoT; Aplicações IoT.

ABSTRACT

The goal of the Internet of Things ecosystem is far beyond devices, cities, objects and infrastructures interconnected and in constant communication. The Internet of Things will integrate and use such elements and data in the development of new services and products that can change and positively impact daily life.

As this is a recent concept and is constantly expanding and changing, technologies, key concepts and developments are systematically reviewed. Therefore, in this thesis one of the primary focus points is to become aware of the state of art knowing that this is a paradigm that incites the interest of influential companies that unite in order to standardize and normalize the Internet of Things.

In a first instance, the concept and applicability of the Internet of Things is discussed, afterwards approaching its principles and paradigms. The investigation of different architectures allows the acquisition of a set of information about the strategies to adopt regarding the Internet of Things, thus composing a map that goes from the protocols to the first frameworks and platforms. The challenges in this paradigm are explored, leading to a discussion perspective that transposes into the future of both hardware and software.

In order to allow a practical evaluation of the aspects related to the Internet of Things, an experiment technology is selected and a case study is developed and the obtained results are reported.

KEYWORDS: INTERNET OF THINGS; CHALLENGES; IOT TECHNOLOGIES; IOT APPLICATIONS.

ÍNDICE

ÍNDICE DE FIGURAS

ÍNDICE DE TABELAS

LISTA DE SIGLAS E ACRÓNIMOS

AMQP - Advanced Message Queuing Protocol

CoAP - Constrained Application Protocol

DDS - Data Distribution Service

DR - Disaster Recovery

GF – Grupo funcional

GUI - Graphical User Interface

HA - Highly-Available

HTTP - Hypertext Transfer Protocol

IoT - Internet of Things

JSF - Java Server Faces

MQTT - Message Queuing Telemetry Transport

OMG - Object Management Group

OSI - Open System Interconnection

OTA - Over-the-air

PEP – Policy Enforcement Point

PHP - Hypertext Preprocessor

QoS – Quality of Service

RA - Referencial Architecture

REST - Representational State Transfer

RMS – Remote Monitoring Station

SaaS - Software-as-a-Service

SAML2 – Security Assertion Markup Language 2.0

SASL – Simple Authentication and Security Layer

SDP – Session Description Protocol

SSN - Semantic Sensor Networks

SSO - Single Sign-On

TLS – Transport Layer Security

1. INTRODUÇÃO

1.1Enquadramento

A Internet das Coisas é o "nome que damos ao que se obteve após 30 anos de evolução convergente na Internet, comunicações sem fio, processadores, memórias, protocolos de comunicações leves, aprendizagem de máquinas e sensores" (Rajkumar Buyya, 2016), afirma *Scot Stelter*, vice-presidente da ChainLink Research.

Uma vez que a internet pode integrar todos os dispositivos, máquinas, objetos, cidades e/ou infraestruturas surge, consequentemente, a necessidade de os interconectar. O cenário das casas inteligentes é um ótimo exemplo de como a Internet das Coisas permite simplificar as tarefas do dia-a-dia, como por exemplo, a possibilidade de programar o horário de abertura de persianas, controlar remotamente o sistema de aquecimento ou, através de uma aplicação, definir quando é que o aspirador deve iniciar o seu trabalho. Esta interação para além de simplificar algumas tarefas, permite maior conforto ao utilizador, poupança de energia, redirecionamento do seu tempo livre para outras atividades e a possibilidade de monitorização de todos os sistemas, antecipando assim possíveis problemas. Assim, a Internet das Coisas emerge como uma combinação de múltiplas tecnologias com diferentes aplicações.

Segundo a Cisco, em 2020 o número de dispositivos conectados atingirá os 50 mil milhões (Callcentermagazine, 2015). Casas, carros e cidades inteligentes serão elementos responsáveis pelo impulso deste mercado, assim como a tendência de uma infraestrutura cada vez mais conectada. Consequentemente surgem os consórcios, ou uniões de empresas mundialmente influentes, que têm como objetivo unir esforços de modo a aumentar a educação IoT entre consumidores, canais de vendas e de investidores. Por outro lado, em termos tecnológicos a união de esforços estende-se também à comunidade académica e a investigadores. Foca-se na promoção e desenvolvimento de normas, definição das melhores estratégias para fluxos de dados, estruturação das melhores plataformas, segurança e proteção de dados. O objetivo final é facilitar a adoção da Internet das Coisas em grande escala.

1.2Motivação

Na medida em que as redes de informação se tornaram parte integrante da vida quotidiana das pessoas, existe atualmente uma enorme necessidade de obter os recursos corretos, tanto para o sucesso pessoal como para o profissional. A Internet das Coisas possibilita a oportunidade de novas funcionalidades de tecnologias que influenciam a estrutura de mercado, criando vantagens e ameaças ao crescimento e desenvolvimento. Tendo todos estes aspetos em consideração, este conceito traz inúmeras vantagens que afetam tudo e todos, em qualquer lugar, a qualquer momento, em qualquer rede, independentemente do serviço em questão.

Trata-se de uma área que começa a dar os primeiros passos, e para a qual existem diversas propostas de normalização e tecnológicas. Contudo não existirá internet das coisas sem haver uma uniformização abrangente, conseguida através de normas e arquiteturas de referência. Interessa por isso estudar as propostas atuais e compreender quais se encontram em melhor posição para serem adotadas. Além disso, plataformas de hardware e software que não sigam normas IoT, ou arquiteturas de referência têm menos hipóteses de serem tecnologias com futuro. Desta forma, interessa também estudar as plataformas atuais desse ponto de vista e averiguar a sua maturidade.

1.3Objetivos

O objetivo central desta dissertação consiste em realizar um estudo sobre a Internet das Coisas. Na secção anterior compreendeu-se que existem inúmeras vantagens tanto para as áreas científicas, industriais e empresariais como as de consumo. Contudo é necessário enfrentar vários desafios e desenvolver soluções conceituais e tecnologias adequadas. Estes desafios incluem o desenvolvimento de arquiteturas adequadas, abandonando os sistemas fechados e adotando sistemas abertos, enfrentando consequentemente os problemas de privacidade, questões de ética, armazenamento de dados, processamento, protocolos de comunicação, objetos inteligentes e descoberta de novos serviços. Por isso, esta dissertação irá focar-se em arquiteturas de referência direcionadas para a Internet das Coisas, analisando posteriormente protocolos relacionados, tendo em vista a análise de tecnologias que sustentaram o conteúdo teórico desta dissertação com resultados e discussões.

De modo a que o objetivo central seja alcançado, foram estabelecidos vários objetivos específicos:

- Identificar os princípios e paradigmas associados à Internet das Coisas;
- Investigação de arquiteturas de referência para a Internet das Coisas;
- Investigação de arquiteturas de *middleware* para a Internet das Coisas;
- Análise de protocolos e plataformas;
- Relações entre as arquiteturas de referência, protocolos e plataformas;
- Desafios associados à Internet das Coisas;
- Tendências futuras: Sistemas *Open Source*, Sistemas operativos incorporados (*Embedded Operating Systems*), Sistemas verticais;
- Soluções/Plataformas
    - Identificação de soluções disponíveis no mercado;
    - Análise de soluções;
    - Demonstração e exposição dos resultados das soluções.

1.4Estrutura do documento

O presente documento encontra-se dividido em seis capítulos, contendo ainda uma secção de anexos.

O primeiro capítulo contém um breve enquadramento ao tema da dissertação, a motivação da mesma, a definição dos objetivos e a descrição da estrutura do documento.

O capítulo dois aborda o estado da arte, começando pela arquitetura e modelo de referência onde é distinguida a diferença entre estes dois conceitos, apresentando-se posteriormente os requisitos de uma arquitetura de referência. Em seguida são analisadas as arquiteturas de referência IoT-A e a WSO2, assim como as plataformas de *middleware* para IoT, analisando, a título de exemplo, o projeto OpenIoT. Posteriormente são analisados e identificados os protocolos mais relevantes, tais como MQTT (*Message Queuing Telemetry Transport*), HTTP (*HyperText Transfer Protocol*), CoAP (*Constrained Application Protocol*), etc, e são analisadas quatro plataformas IoT. Com base na informação adquirida é estabelecida a relação entre arquiteturas de referência, protocolos e plataformas

No capítulo três faz-se uma análise à abordagem das soluções/plataformas selecionadas e é elaborado um plano de testes para as mesmas.

O capítulo quatro aborda o desenvolvimento, ou seja, a configuração/programação das soluções assim como os testes realizados às mesmas.

No quinto capítulo, são apresentados os resultados obtidos provenientes dos cenários de teste.

No último capítulo são expostas as conclusões obtidas ao longo desta dissertação e delineadas algumas indicações para trabalho futuro.

2. ESTADO DA ARTE

No estado da arte do presente documento, são abordadas seis importantes secções que permitem estabelecer a ligação com o restante conteúdo abordado posteriormente. A primeiro secção faz a distinção entre arquiteturas e modelos de referência para IoT, onde se destaca a arquitetura de referência IoT-A e a WSO2. Na secção seguinte, é destacada uma plataforma de *middleware* para IoT, o projeto OpenIoT. Posteriormente são abordados os protocolos atualmente mais relevantes para a comunicação em IoT. Sendo que posteriormente foram selecionadas quatro soluções/plataformas (Projeto Kaa, Ubiquitousware - Fujitsu, Amazon IoT e MpDS), das quais apenas a solução Intelligent Care e Worker Safety da Fujitsu, e a solução MpDS foram testadas. Por último, foi estabelecida a relação entre arquiteturas de referência, protocolos e plataformas IoT.

2.1Arquitetura e modelo de referência

2.1.1 Conceito de arquitetura e modelo de referência

Uma arquitetura de referência pode ser entendida como uma arquitetura abstrata que envolve conhecimento e experiências sobre como projetar sistemas num determinado domínio, acompanhando assim o seu desenvolvimento e evolução. Os termos arquitetura de referência (AR) e modelo de referência (MR) têm sido utilizados como sinónimos, contudo possuem definições distintas. Um modelo de referência é um artefacto abstrato que apresenta um conjunto de conceitos comuns e relacionamentos entre eles com relação a um domínio específico, sendo independentes de padrões, tecnologias, implementações, entre outros. Por outro lado, uma arquitetura de referência pode ser concebida com base em um ou mais modelos de referência, de modo a especificar de maneira unificada e não ambígua as regras de negócio, estilos ou padrões arquiteturais, decisões arquiteturais, boas práticas de desenvolvimento e elementos de hardware e/ou software necessários à criação de arquiteturas concretas. Ou seja, um modelo de referência é usado para regular a base comum a ser adotada para estabelecer uma arquitetura de referência, que por sua vez, fornece os elementos concretos e abstratos que devem ser tidos em conta para conceber a

arquitetura de sistema. (Paulo F. Pires, 2015). A figura 1 retrata os relacionamentos entre o modelo de referência, arquiteturas de referência e arquiteturas concretas.

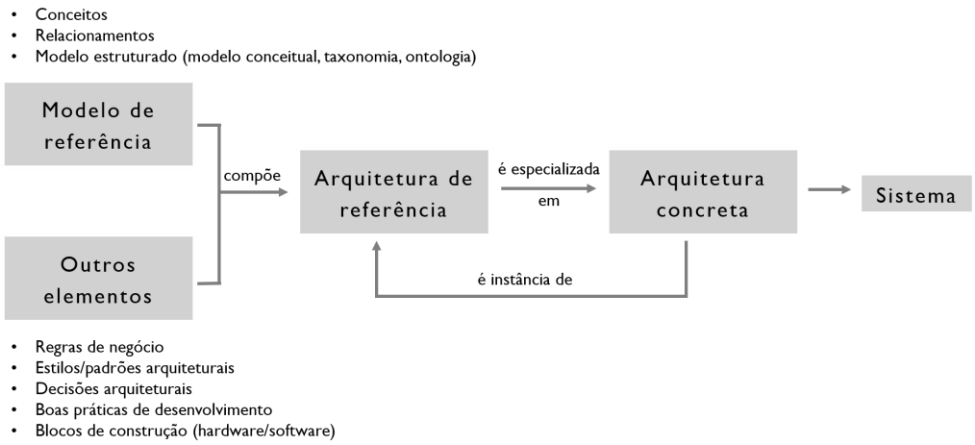


Figura 1 – Relacionamentos entre modelos de referência, arquitetura de referência e arquitetura concreta. (Adaptada de (Paulo F. Pires, 2015)).

Numa arquitetura de referência destacam-se os seguintes objetivos (Paulo F. Pires, 2015):

- **Facilitar o desenvolvimento de sistemas, promovendo a redução de tempo e custo**, ou seja, uma AR é capaz de influenciar diretamente a produtividade e a qualidade do desenvolvimento, principalmente por providenciarem os elementos fundamentais para a construção das arquiteturas concretas desses sistemas.
- **Padronizar arquiteturas de sistemas num domínio**, ou seja, através do consenso estabelecido por uma AR ao nível dos elementos fundamentais a serem considerados e diretrizes a serem seguidas, é possível desenvolver arquiteturas concretas interoperáveis entre si, facilitando a integração e a compatibilidade entre diferentes sistemas heterogéneos no domínio em questão. Evitando, deste modo, a decadência da arquitetura.
- **Adaptar a evolução de sistemas existentes**, ou seja, torna-se essencial nos sistemas atualmente em execução haver uma modificação das suas funcionalidades com o objetivo de atender a novos requisitos, promover melhorias de qualidade ou adaptar as necessidades a novos contextos.

2.1.2 Requisitos arquitetura de referência

De modo a desenvolver-se uma arquitetura de referência para IoT, é necessário ter em consideração alguns requisitos específicos, que são exclusivos dos dispositivos IoT e dos ambientes que os suportam. Dado que o número de dispositivos conectados tende a aumentar rapidamente nos próximos anos é essencial uma arquitetura adequada à escalabilidade, com uma abordagem *Highly-Available* (HA), que suporte a implantação de centros de dados para permitir *Disaster Recovery* (DR). Outros requisitos provêm da forma como os dispositivos são fabricados e utilizados. Deste modo é possível categorizar os requisitos por: conectividade e comunicação, gestão de dispositivos, processamento de dados, escalabilidade, segurança, análise preditiva, integração e HA.

Ao nível da conectividade e comunicação, os protocolos existentes, como o HTTP, assumem um lugar de destaque para muitos dispositivos, fornecendo uma conectividade universal. Contudo os protocolos tradicionais da internet, podem representar um problema, essencialmente por dois motivos principais: o tamanho da memória do programa necessário assim como os requisitos de energia, podem ser incorporáveis em dispositivos com poucos recursos. Daí ser conveniente, em termos de viabilidade económica utilizar protocolos simples, com mensagens menores e binárias.

A gestão ativa de dispositivos tornou-se um recurso essencial que permite efetuar inúmeras atividades de relevo, tais como, a capacidade de desconectar, localizar e eliminar dados de um dispositivo roubado ou perdido, possibilidade de atualizar software e credenciais de segurança, ativação ou desativação remota de determinados recursos de hardware.

Uma arquitetura de referência tem de ser projetada para suportar uma grande quantidade de dispositivos, que consequentemente produzem um enorme volume de dados. Inclui assim, o requisito de um sistema de armazenamento altamente escalável, que permite lidar com diversos dados e volumes elevados. Além disso, em vários casos, como por exemplo o das cidades inteligentes, os dispositivos precisam de ser capazes de analisar e atuar sobre os dados em tempo real. Em alguns dispositivos, a lógica será simples e incorporada enquanto noutros, mais robustos e complexos, os mecanismos serão mais inteligentes para processamento e atuação.

Idealmente qualquer arquitetura do lado do servidor seria altamente escalável e capaz de suportar milhões de dispositivos, todos constantemente a enviar, receber e a tratar os dados. Contudo, o custo associado tanto ao hardware como ao software é elevado. Deste modo, um requisito importante é oferecer suporte ao dimensionamento de uma pequena implantação para um número muito grande de dispositivos. A escalabilidade elástica e a capacidade de implantar uma infraestrutura de cloud são essenciais, permitindo alojar os serviços e distribuí-los por servidores de menor custo.

A segurança é atualmente um dos aspetos que mais preocupa os investigadores que se centram neste conceito. Os dispositivos podem exercer ações críticas que se traduzem em duas categorias de riscos:

- Riscos inerentes a qualquer sistema da internet, como por exemplo portas de protocolo abertas nos dispositivos;
- Riscos específicos exclusivos dos dispositivos IoT, que inclui questões singulares relativas ao hardware. Por exemplo, muitos dos dispositivos IoT são demasiado pequenos para suportar criptografia assimétrica[1], o que dificulta a proteção das comunicações.

Após a distinção entre arquitetura de referência e modelo de referência bem como a análise e compreensão dos requisitos essenciais para uma arquitetura de IoT, a literatura apresenta soluções que pretendem colmatar falhas que as arquiteturas não destinadas a este conceito possam enfrentar.

O projeto IoT-A (Paulo F. Pires, 2015) propõe um modelo arquitetural de referência (MAR), que envolve uma AR base e a definição de um conjunto de características chave para a sua construção. Esta AR é definida num nível de abstração elevado fornecendo visões e perspetivas relevantes.

O MAR fornece as visões, (i) funcional, (ii) informação, (iii) operação, e (iv) implantação. Além das visões, o modelo define também perspetivas arquiteturais que abordam interesses comuns a mais do que uma visão. Interesses esses que estão relacionados com requisitos não funcionais ou atributos de qualidade. Uma perspetiva arquitetural é definida com uma coleção de atividades, táticas e diretrizes que são usadas

[1] Também conhecida como criptografia de chave pública, é uma classe de protocolos de criptografia baseados em algoritmos que requerem duas chaves, uma delas secreta e a outra pública.

para garantir que um sistema exiba um conjunto particular de atributos de qualidade relacionados. O MAR fornece as seguintes perspetivas: (i) evolução e interoperabilidade; (ii) disponibilidade e resiliência; (iii) confiabilidade, segurança e privacidade, e; (iv) desempenho e escalabilidade.

A **visão funcional** (VF), tal como o nome indica, fornece as principais funcionalidades a serem consideradas no projeto de um *middleware* IoT. Conforme ilustrado na figura 2, a VF possui nove grupos de funcionalidades: (i) aplicação; (ii) gestão; (iii) organização de serviços; (iv) gestão de processos IoT; (v) entidade virtual; (vi) serviço IoT, (vii) segurança, (viii) comunicação; e; (ix) dispositivo. Cada um destes grupos funcionais (GFs) envolve um ou mais componentes funcionais (CFs), representados na figura por retângulos de fundo branco. Contudo apesar da visão funcional descrever os componentes funcionais, não especifica as interações que ocorrem entre estes elementos, pelo facto de as mesmas serem tipicamente dependentes das escolhas de projetos, sendo, portanto, realizadas durante o desenvolvimento da arquitetura concreta.

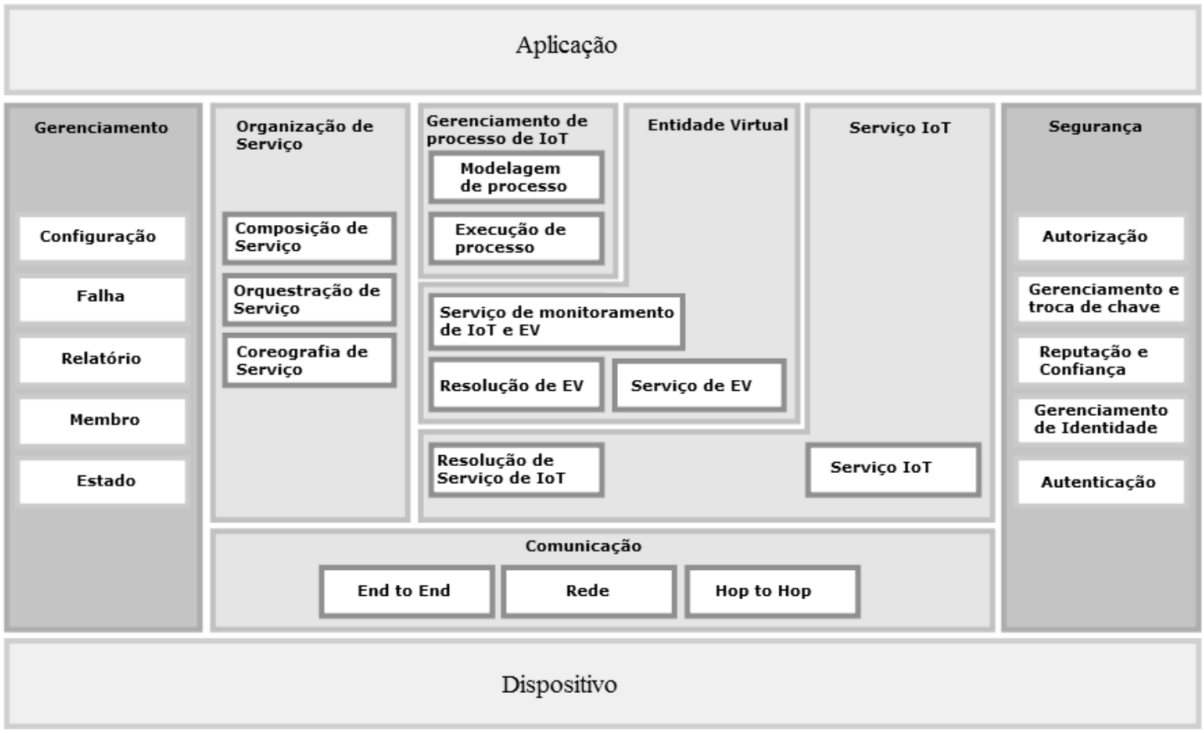


Figura 2 - Decomposição da visão funcional da arquitetura de referência IoT-A.

Antes de se analisar os diversos elementos que compõem a VF é importante esclarecer alguns aspetos sobre as Entidades Virtuais (EVs). Uma Entidade Virtual (EV)

representa/modela uma entidade física (objeto), sendo composta por um identificador, por um tipo (*entityType*) e por uma serie de atributos que a descrevem. Através desses atributos obtêm-se as funcionalidades de uma EV e, inclusive, é pela modificação dos mesmos que é possível gerar estímulos/atuar sobre o objeto físico que ela representa. As EVs possuem dois tipos de classificação: ativa ou passiva. A primeira é aquela que permite aceder a recursos do ambiente sendo composta por aplicações, agentes ou serviços. Em oposição as EVs passivas representam elementos que compõem os recursos do sistema como, por exemplo, registos em bases de dados. As EVs interagem com os serviços através de associações que indicam qual a informação ou ação que pode ser respetivamente obtida ou realizada sobre uma EV.

O **GF Gestão de Processos** tem como objetivo promover os principais conceitos e interfaces necessárias para adaptar as convencionais gestões de processos, às particularidades dos ambientes IoT. Para tal, as principais tarefas estão divididas em dois CFs: modelação de processos e execução de processos. A primeira fornece um conjunto de ferramentas que permitem descrever e modelar os vários processos de um ambiente IoT. Já o CF execução de processos, é responsável por implementar os processos de gestão de serviços IoT de forma a que as tarefas a serem executadas sejam alocadas a ambientes de execução apropriados. Para tal, o CF Execução de Processos interage com o GF Organização de Serviços.

O **GF Organização de Serviços** atua como um ponto central de comunicação dos diferentes GFs, onde se realiza a composição, orquestração e coreografia de serviços. Para que tal aconteça, este GF recebe os requisitos necessários das tarefas IoT, localiza os serviços IoT adequados à sua execução e invoca-os, sendo estas funcionalidades executadas pelos três CFs: (i) Composição de Serviço; (ii) Orquestração de Serviço; e; (iii) Coreografia de Serviço. O primeiro CF, cria serviços mais elaborados a partir de serviços IoT básicos. O segundo CF, é responsável por controlar e coordenar os serviços IoT adequados de forma a responder às requisições provenientes de outros CFs e dos utilizadores. Por fim, o CF Coreografia de Serviço, cuja função é atuar como um *broker* para que os serviços IoT interajam entre si utilizando um modelo *publish-subscriber*. Assim, deste modo caso não estejam disponíveis imediatamente os serviços, a requisição é memorizada e o cliente é notificado quando o serviço voltar a estar disponível.

O **GF Entidade Virtual** consiste em três CFs: (i) Resolução de EV; (ii) Serviço de EV; e; (iii) Serviço de Monitorização de IoT e EV. O **CF Resolução de EV** permite que o utilizador

recupere associações entre EVs e serviços IoT. O **CF Serviço de EV** manipula serviços de entidades, que representa um ponto de acesso global para uma entidade especifica oferecendo meios para aprender e manipular o seu estado. O **CF Serviço de Monitorização de IoT e EV** é responsável por encontrar automaticamente novas associações que estão incluídas no CF Resolução de EV.

O **GF Serviço IoT** é composto pelos CFs: (i) Serviço IoT; e; (ii) Resolução de Serviço de IoT. O CF Serviço IoT é responsável por tornar um recurso disponível para o restante sistema, para tal, as suas principais funções são: (i) retornar informações mantidas por um recurso IoT mediante requisições síncronas; (ii) receber informações a serem armazenadas em recursos IoT, assim como enviar dados para agir sobre atuadores e para parametrizar a configuração de recursos; (iii) fornecer informações de forma assíncrona utilizando um modelo de subscrição. O CF Resolução de Serviços fornece as funcionalidades necessários para se localizar um serviço IoT e informar de como este deve ser utilizado. Deste modo, este CF fornece mecanismos para descrever serviços e para armazenar essas descrições em bases de dados. Tal como um serviço de resolução tradicional, este CF providencia meios para realizar descoberta, *lookup,* resolução de nomes e ainda possui a funcionalidade para criar, modificar e remover descrições de serviços.

O **GF Comunicação** cria uma abstração para diferentes tecnologias de comunicação existentes a partir de três componentes funcionais (CFs): (i) Hop-to-Hop; (ii) Rede; e; (iii) End-to-End. O **CF Hop-to-Hop** é o primeiro nível de abstração tendo como função realizar a comunicação entre os outros CFS, independentemente da camada de enlace existente. Assim, este CF envia e recebe dados tanto do CF Rede como dos dispositivos físicos. Já o **CF Rede**, de forma semelhante à camada de rede do modelo OSI[2], permite a comunicação entre redes distintas através de um esquema de endereçamento (*locators*) e da resolução de identificadores. Este CF recebe e encaminha dados para o CF End-to-End. Por último, o **CF End-to-End**, que providencia a comunicação com o GF Serviço de IoT e oferece, através de argumentos, uma série de características à conexão, tais como, confiabilidade, integridade e cifragem. Além disso, o CF End-to-End possibilita a definição de gateways para a tradução entre diferentes protocolos end-to-end, por exemplo, HTTP/TCP e CoAP/UDP.

[2] É um modelo de rede de computador referência da ISSO dividido em camadas de funções com o objetivo de ser um padrão para protocolos de comunicação entre os mais diversos sistemas numa rede local, garantindo a comunicação entre dois sistemas computacionais (*end-to-end*).

O **GF Segurança** é responsável por garantir a segurança e a privacidade dos sistemas compatíveis com a IoT-A. Este GF está organizado em cinco CFs: (i) Autorização; (ii) Gestão e Troca de Chave; (iii) Reputação e Confiança; (iv) Gestão de Identidade; e; (v) Autenticação. O **CF Autorização** é um *front-end* para gerir e executar decisões de controlo de acesso baseadas em políticas. O **CF Gestão e Troca de Chave** habilita comunicações seguras entre dois ou mais pares que inicialmente não se conhecem ou cuja a interoperabilidade não esteja garantida, assegurando integridade e confiabilidade. Ou seja, este CF possui duas funcionalidades, distribuir chaves de forma segura e registar recursos de segurança. O **CF Reputação e Segurança** reune a pontuação de reputação de utilizadores e calcula os níveis de confiança do serviço. O **CF Gestão de Identidade** lida com questões de privacidade, emissão e gestão de pseudónimos e informação extra para que os componentes confiáveis possam operar de forma anonima. O **CF Autenticação** é responsável pela autenticação de serviços e utilizadores, verificação de credencias fornecidas pelo utilizador e, caso estas sejam válidas, devolução de uma assertiva como resultado.

O **GF Gestão** (representado na figura 2 por "Gerenciamento") consiste em cinco CFs: (i) Configuração; (ii) Falha; (iii) Relatório; (iv) Membro e; (v) Estado. O **CF Configuração** é responsável por realizar funções de inicialização da configuração do sistema, tais como coletar e armazenar as configurações dos demais CFs e dispositivos. Sendo também responsável por rastrear as mudanças de configuração e realizar planeamento para futuras extensões do sistema. O **CF Falha** tem como objetivo identificar, isolar, corrigir e registar falhas que ocorrem no sistema IoT. Para cada ocorrência de falha é enviada uma notificação pelo CF correspondente para o CF Falha, o qual reúne mais dados a fim de identificar a natureza e o grau do problema. O **CF Relatório** permite refinar as informações fornecidas pelos outros CFs, produzindo relatórios ou recuperando relatórios de um histórico. O **CF Membro** é responsável pela gestão de associações a membro e informações importantes de qualquer entidade relevante. Este CF possui três funções principais: monitorização continua de membros, recuperação e atualização de membros. O **CF Estado** visa monitorizar e fornecer os estados passado, presente e futuro do sistema IoT que são requeridos pelo CF Falha.

A **Visão da Informação** (VI) tem como foco a descrição, o tratamento e o ciclo de vida da informação, bem como o seu fluxo entre todos os componentes do sistema. De modo a

descrever-se essas interações, a arquitetura IoT utiliza a abstração fundamental das Entidades Virtuais (EVs).

A **Visão de Operação e Implantação** (VOI) tem como objetivo primordial fazer com que os sistemas atuais comuniquem e operem de uma forma abrangente, isto é, com o maior número de sistemas. Os pontos de vista utilizados nesta visão são:

- **Diagrama de Modelo do Domínio** que é utilizado como uma diretriz para descrever uma aplicação específica de domínio.
- **Modelo funcional** que é utilizado como referência para a definição do sistema. Em particular definem-se grupos funcionais tais como serviços de IoT e grupos de conectividade, que são fundamentais para uma definição correta do sistema.
- **Diagramas de Conectividade de Rede** que podem ser utilizados para planear a topologia de conectividade, de modo a habilitarem-se as capacidades de rede desejadas para a aplicação alvo. Ao nível da implantação, este diagrama é utilizado para definir hierarquias e o tipo de sub-redes formando assim um sistema completo.

2.1.3 Arquitetura de Referência WSO2

A figura 3 ilustra a arquitetura de referência WSO2 que consiste num conjunto de camadas, sendo duas delas transversais às outras. As camadas são: (i) Comunicações (Communications); (ii) Processamento de Eventos e Análise (Event Processing and Analytics); (iii) Camada de Agregação/Barramento (Agregation/Bus Layer); (iv) Comunicações entre Dispositivos (Devices); e as duas transversais (v) Gestão de Dispositivos (Device Manager); e; (vi) Gestão de Acesso e Identidade (Identity & Access Management).

A **camada** inferior é composta pelos dispositivos físicos **(*Devices*)**, que de modo a serem considerados dispositivos IoT devem possuir uma comunicação direta ou indireta com a Internet. No modo de comunicação direta, o dispositivo possui no seu hardware uma interface de comunicação, como é exemplo o WiFi ou Ethernet. Por sua vez, na forma indireta, o dispositivo requer um dispositivo intermediário para estabelecer a conexão com a Internet, como por exemplo, as etiquetas RFID ou Bluetooth. Para além da capacidade de conexão é recomendado que um dispositivo possua um identificador único universal (UUID) - local ou global - e um *token OAuth2 Refresh/Bearer* como identificador auxiliar.

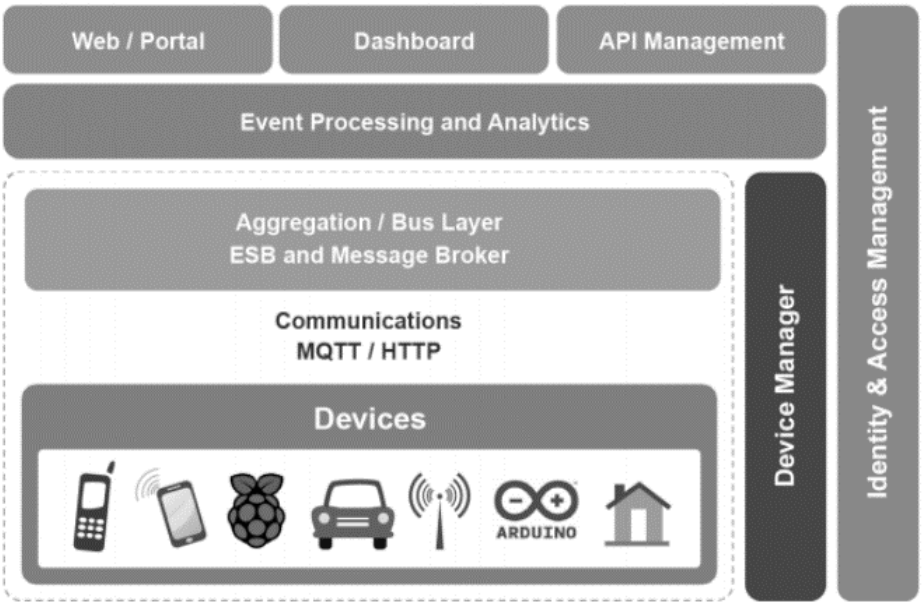


Figura 3 - Arquitetura de referência para IoT da WSO2(Retirada de (WSO2, 2015)).

A **Camada de Comunicações (*Communications*)** suporta a conectividade dos dispositivos. A proposta indica o uso de três protocolos: (i) MQTT (secção 2.3.1); (ii) HTTP (secção 2.3.2) e o seu uso no estilo arquitetural REST (Representational State Transfer) e CoAP (secção 2.3.3).

A **Camada de Agregação/Barramento (*Aggregation/Bus*)** assume um papel importante por três razões. Em primeiro lugar porque tem a responsabilidade de implementar um servidor HTTP, ou um *broker* MQTT, de modo a permitir a comunicação com os dispositivos. Segundo, pela capacidade de agrupar e combinar informações de diferentes dispositivos, encaminhando-os para um dispositivo específico. E, por último, por realizar o papel de *gateway* convertendo mensagens MQTT para pedidos HTTP, e vice-versa, servindo como um meio de ligação entre dispositivos que falam diferentes protocolos. Além destas funções, esta camada tem de um papel chave na segurança. Para tal tem de atuar como servidor *OAuth2*, validando os *bearer Tokens* e realizando o controlo de acessos baseado em políticas (PEP - *Policy Enforcement Point*).

A **Camada de Processamento de Eventos e Análise (*Event Processing and Analytics*)** é responsável por obter informações (eventos) da camada de agregação/barramento, processá-los e agir sobre eles. Um dos pontos chave desta camada é o armazenamento de dados em bases de dados que podem (não) ser do tipo relacional, com comunicação servidor-cliente como é exemplo uma aplicação RESTful, ou abordagens com mais funcionalidades como é o caso das plataformas analíticas de *Big Data*. Neste último caso, a

plataforma consiste numa cloud escalável com suporte a tecnologias como *Apache Hadoop*[3] para promover análises baseadas em *map-reduce* sobre os dados. A camada pode suportar plataformas de processamento de aplicações tradicionais tais como JavaBeans[4] ou alternativas como node.js, PHP (*Hypertext Preprocessor)*, Ruby ou Phyton.

Por fim, as duas camadas transversais, **gestão de dispositivos e gestão de identidade e acesso**. O objetivo da primeira é munir-se de mecanismos que permitam a configuração remota dos dispositivos, como a instalação de imagens de sistemas, o suporte à realização de inventários, a monitorização do funcionamento do dispositivo, o fornecimento de informações sobre a sua localização e disponibilidade, e, auxiliar na determinação de controlos de segurança e identidade. Estes mecanismos são implementados com o auxílio de dois componentes: um servidor e um agente que é executado no próprio dispositivo.

A camada de **Gestão de Identidade e Acesso (*Identity and Access Management*)** é responsável por promover mecanismos de autenticação e autorização para que seja possível aceder aos recursos. Para tal, a proposta WSO2 recomenda a emissão e validação do *token* OAuth2, outros serviços de identidade incluindo suporte a SAML2 SSO[5] e OpenID Connect[6] de modo a identificar requisições de entrada na camada Web, eXtensible Access Control Markup Language Policy Decision Point (XACML PDP), diretório de utilizadores e gestão de políticas para controlo de acesso.

2.2Plataformas de *Middleware* para IoT

Uma plataforma de *middleware* é um artefacto de *software* que reside entre a camada de aplicação e a infraestrutura de suporte (comunicação, processamento, sensoriamento), fornecendo acesso padronizado aos dados e serviços munidos pelos objetos inteligentes, através de interfaces de alto nível. Além disso, promovem a reutilização de serviços genéricos, que podem ser compostos e configurados para facilitar o desenvolvimento de aplicações de forma mais eficiente para o ambiente IoT.

[3] Plataforma de *software opensource* para armazenamento e processamento distribuído de grandes quantidades de dados (Cetax, 2017)
[4] Componentes de software escritos na linguagem de programação Java.
[5] Versão do padrão SAML para a troca de dados de autenticação e autorização entre domínios de segurança
[6] Camada de identidade sobre o protocolo OAuth2

2.2.1 OpenIoT

Os autores do projeto OpenIot[7], afirmam que ainda não existe uma maneira fácil de integrar serviços heterogêneos geográfica e administrativamente dispersos, e serviços IoT de forma semanticamente interoperável. Em (Rajkumar Buyya, 2016) é apresentada uma visão geral do projeto que fornece uma plataforma de *middleware*, permitindo unificar semanticamente diversas aplicações IoT na *cloud*. O OpenIoT usa a ontologia W3C SSN (Semantic Sensor Network Ontology), que serve como base para a especificação de solicitações dos serviços, combinando sensores, fluxos de dados e as suas propriedades. A arquitetura OpenIoT compreende três planos lógicos distintas apresentadas na figura 4: (i) Utilidade/Aplicação; (ii) Virtual; e; (iii) Físico.

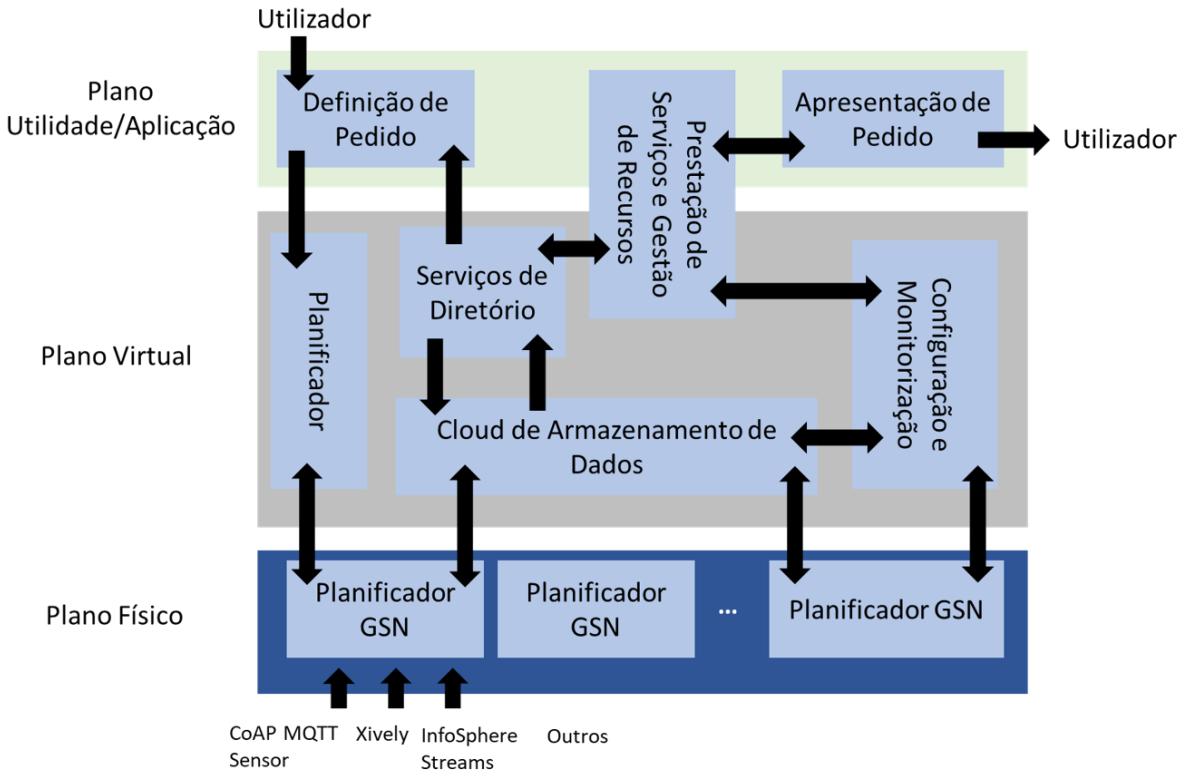


Figura 4 -Arquitetura da plataforma de middleware OpenIoT (Adaptada de *(Rajkumar Buyya, 2016)*.

O plano **Utilidade/Aplicação** é composto por três módulos: (i) Definição de Pedido; (ii) Apresentação de Pedido; (iii) Prestação de Serviços e Gestão de Recursos. Através de uma interface Web, o primeiro módulo, **Definição de pedido** permite definir solicitações de

[7] FP7-287305 co-financiado pela Comissão Europeia

serviços para a plataforma em tempo de execução, solicitações essas que são enviadas para o Agendamento. O módulo **Apresentação de Pedido** possibilita, também via interface Web, que o utilizador selecione *mashups* de um repositório. Por último, o módulo de **Prestação de Serviços e Gestão de Recursos** tem como função combinar fluxos de dados para serem entregues aos serviços solicitantes e realizar a contabilidade e a faturação dos recursos da plataforma.

O plano **Virtual** é composto por quatro módulos: (i) Cloud de Armazenamento de Dados; (ii) Serviço de Diretório; (iii) Planificador; e; (iv) Configuração e Monitorização. A **Cloud de Armazenamento de Dados** atua como base de dados que permite o armazenamento de fluxos de dados decorrentes do *middleware dos sensores*. O módulo **Serviço de Diretório** armazena as informações sobre todos os sensores que estão disponíveis na plataforma OpenIoT. Também fornece os serviços para registar sensores assim como, para efetuar a sua descoberta/pesquisa. O módulo **Planificador** processa pedidos de implementação provenientes de serviços e examina o acesso adequado aos recursos. E por fim, o módulo **Configuração e Monitorização** permite a gestão e a configuração dos dispositivos conectados e dos serviços em execução na plataforma.

Por último, o plano **Físico** é apenas composto pelo módulo **Sensor *Middleware***, responsável por colecionar, filtrar e combinar fluxos de dados de outros sensores virtuais ou dispositivos físicos. Para tal, utiliza uma versão adaptada do *middleware* GSN (Global Sensor Network), que fornece uma API RESTful para a interoperação com os dispositivos, bem como alguns recursos de segurança.

Esta arquitetura não especifica tecnologias de implementação associadas aos vários componentes, fornecendo assim uma apresentação abstrata dos elementos funcionais da arquitetura. Contudo, o OpenIoT tem sido implementado com base em tecnologias específicas, como GSN para o *middleware* do sensor, W3C SSN para o serviço de diretório e as bibliotecas JSF, como *Primefaces*[8]. Todavia, implementações baseadas em tecnologias alternativas são possíveis. Os serviços fornecidos através da plataforma, dependem de dados coletados e transmitidos para a *cloud* através do *middleware* do sensor (GSN). Dada a existência de vários fluxos de dados na cloud, um fluxo de trabalho típico associado ao uso da plataforma OpenIot envolve (Rajkumar Buyya, 2016):

[8] Framework open source para Java Server Faces.

- A formulação de um pedido para um serviço de IoT utilizando a ferramenta definição de pedido e o seu envio ao componente planificador (*scheduler) global*. O pedido específica os sensores envolvidos e o tipo de processamento a ser aplicado sobre os dados, bem como a visualização preferencial dos resultados.
- A análise do pedido de serviço de IoT pelo planificador (*scheduler)* e a descoberta dos sensores envolvidos para fornecer o serviço de IoT. Para descobrir os sensores utiliza-se o serviço de diretório.
- A formulação do serviço (por exemplo, na forma de uma consulta SPARQL) e a sua persistência na cloud, juntamente com outros metadados sobre o serviço. Os metadados incluem um identificador para o serviço IoT criado.
- A execução do serviço pelos utilizadores finais (com base na manipulação do serviço de destino) e a visualização dos resultados.

Em suma, o OpenIoT fornece uma plataforma para a convergência IoT/Cloud que permite a integração de dados e aplicações IoT em infraestruturas de computação em *cloud*, implantação e acesso seguro a aplicações semanticamente interoperáveis, manuseamento de sensores móveis e parâmetros de QoS associados.

2.3Protocolos

A padronização da comunicação é um dos aspetos mais importantes para o desenvolvimento da IoT. Os protocolos para IoT são categorizados em: dispositivo a dispositivo (D2D), dispositivo a servidor (D2S) e servidor a servidor (S2S). Os protocolos D2D servem para interconectar dispositivos diretamente entre si garantindo um conjunto de requisitos como: tempo real, garantias de entrega, alto desempenho, etc. Já os protocolos D2S são destinados a coletar dados e a enviar para sistemas externos (servidores). Por último, os protocolos S2S são utilizados para gerir e integrar informações entre servidores, ou seja, estão no nível de gestão de controlo.

2.3.1 MQTT

"O MQTT foi desenvolvido por *Andy Stanford-Clark* (IBM) e *Arlen Nipper* (Eurotech - agora Cirrus Link) em 1999" (HiveMQ, 2015) com o objetivo de ter um protocolo que fosse

eficiente em termos de largura de banda e consumisse pouca bateria, porque os dispositivos encontravam-se conectados via satélite.

O protocolo é baseado no padrão de troca de mensagens *publish/subscribe* em contraste com o protocolo HTTP que se baseia em mensagens *request/response*. A comunicação *publish/subscribe* é orientada a eventos e permite que as mensagens sejam enviadas para os clientes finais.

A figura 5 demonstra um fluxo de mensagens entre os remetentes e os recetores, em que o MQTT Broker é o ponto central da comunicação e o responsável pelo envio de mensagens com dados aos recetores. Quando um remetente publica uma mensagem para o MQTT Broker, inclui um tópico de mensagem, neste caso o tópico é "temperatura". Este tópico permite identificar as mensagens, e deste modo cada recetor que pretenda receber mensagens deste tópico deve subscrever o mesmo. Assim não é necessário que os remetentes se contactem.

Esta arquitetura apresenta soluções altamente escaláveis, sem dependências entre os produtores de dados e os consumidores dos mesmos.

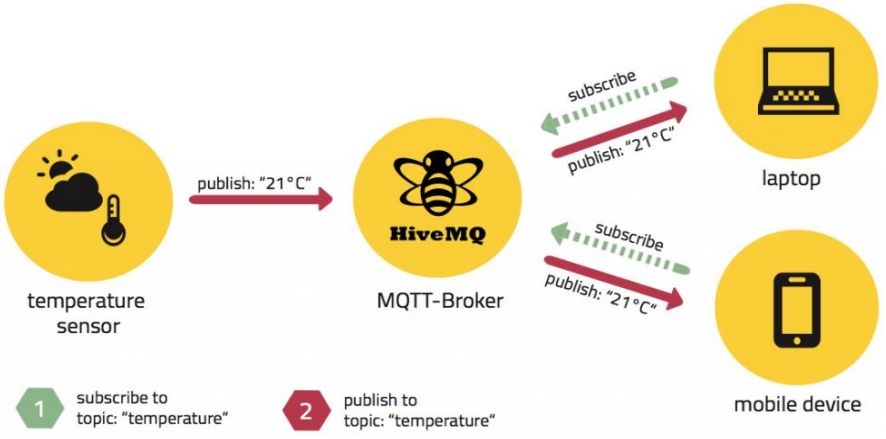


Figura 5 – Exemplo arquitetura publish/subscribe (Retirada de (Stabke Kernel, 2017)).

2.3.2 HTTP

A web evoluiu muito desde que surgiu o HTTP versão 1.1 (HTTP/1.1) em 1999, assim como os requisitos de escala, tempo real, segurança e desempenho. O novo protocolo HTTP versão 2 (HTTP/2) não foi projeto diretamente para IoT, contudo os criadores tiveram em

conta algumas necessidades da mesma. Esta nova versão permite respostas multiplexadas, ou seja, permite enviar respostas em paralelo. Corrigindo o problema de bloqueio do HTTP/1.x onde um pedido pode ficar pendente numa conexão TCP/IP de cada vez. Além disso, promove que clientes e servidores usem uma única conexão TCP na qual os pedidos e respostas são enviadas em fluxos (streams) de forma multiplexada. Este é um recurso importante na medida em que promove um uso mais eficiente das conexões, reduzindo a sobrecarga do HTTP, traduzindo-se assim em transmissões mais rápidas.

O HTTP/2 introduz cabeçalhos compactados usando um formato de compactação de memória muito eficiente, o HPACK (*Header Compression for* HTTP/2). Reduzindo o tamanho de cada solicitação e resposta HTTP, tal como se descreve na figura 6. Ao contrário do HTTP/1.1 que utiliza um protocolo ASCII para transmitir os caracteres, o HTTP/2 utiliza uma codificação binária, ou seja, transmite fluxos binários de dados.

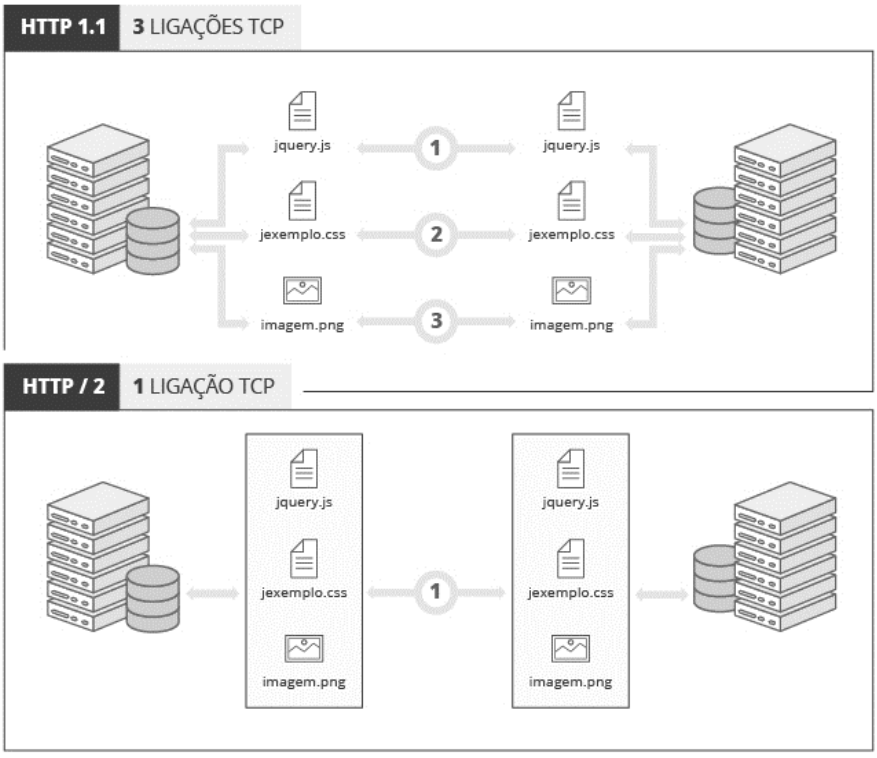


Figura 6 – Ligações TCP HTTP 1.1 Vs HTTP / 2 (Retirada de (Medium, 2018)).

Em virtude dos protocolos binários serem mais eficientes de analisar e mais compactos, os dispositivos com RAM limitada têm maior viabilidade de utilizarem o HTTP/2. O HTTP/2 introduz também o mecanismo de *push* do servidor. Concretamente, significa que

o servidor pode fornecer conteúdo aos clientes sem ter de esperar um pedido. Em suma, o HTTP/2 apresenta as seguintes vantagens em relação ao HTTP/1.1:

- Tempos de carregamentos mais rápidos, já que elimina muitos dos impedimentos do protocolo.
- Mais seguro, dado que o HTTP/2 tem a criptografia ativada por padrão.
- Compatibilidade com dispositivos móveis, pois o recurso de compactação do cabeçalho permite que sites para dispositivos moveis com grande quantidade de pedidos evitem o *download* de *megabytes* desperdiçados dos cabeçalhos
- Menor dependência de *hacks* graças ao recurso de multiplexação. Os métodos que consomem muito tempo para reduzir o número de solicitações do servidor, tais como fragmentação de domínio, *image sprites* ou JavaScript e CSS de preenchimento, não são tão indispensáveis.
- Compatibilidade com o HTTP/1.1, ou seja, servidores e navegadores que permanecem com o HTTP/1.1 podem comunicar com os servidores e navegadores HTTP/2.

2.3.3 CoAP

O CoAP é um protocolo para comunicação com sensores e outros dispositivos com poucos recursos computacionais, que segue um modelo cliente-servidor e é baseado na filosofia REST. No CoAP os servidores disponibilizam recursos por meio de um URL e os clientes acedem a esses recursos usando métodos pré-definidos (POST, GET, PUT, DELETE). Como o CoAP foi desenhado para ser facilmente traduzido em HTTP e este também suporta o modelo REST, podem facilmente ser integrados usando *proxies*, de forma que, por exemplo, uma aplicação cliente pode nem ficar a saber que está a aceder a um sensor.

2.3.4 DDS

DDS é um *middleware* do OMG (Object Management Group) para troca/distribuição de dados no padrão publish-subscribe. Tem como objetivo integrar os componentes de um sistema de modo a fornecer conectividade de dados de baixa latência, extrema confiança e uma arquitetura escalável que os negócios e as aplicações IoT necessitam (OMG (Object Management Group), 2019).

O DDS lida com o endereçamento de mensagens, empacotamento de dados, entrega, controle de fluxo e novas tentativas. Pode fornecer dados seguros com elevada velocidade para milhares de destinatários com controle rigoroso de tempo, confiabilidade, *failover* e heterogeneidade (arquitetura CPU, linguagem de programação e independência do SO (Sistema Operativo)). O DDS suporta uma arquitetura descentralizada que permite a partilha contínua de dados entre *publishers* e *subscribers (figura 7)*. Pode utilizar diferentes protocolos de transporte, incluindo TCP IP e UDP, e a reduzida dimensão da implementação permite que o DDS seja utilizado por dispositivos profundamente embutidos. O DDS está também a surgir como um protocolo de mensagens interoperável para conectar redes de dispositivos em tempo real a *data centers* baseados em cloud.

Figura 7 - Arquitetura do protocolo DDS (Retirado de *(RF Wireless World, s.d.)*).

2.3.5 AMQP

O AMQP é um padrão *open source* para a transmissão de mensagens comercias entre aplicações ou organizações. O protocolo binário oferece autenticação e criptografia através de SASL[9] ou TLS[10], sendo o protocolo de transporte o TCP. Concebido por "JP Morgan em meados de 2006" (Agrawal, 2010), este protocolo de mensagens é rápido, eficiente e multicanal, oferecendo entrega garantida com reconhecimento de mensagens recebidas.

[9] *Framework* para autenticação e segurança de dados nos protocolos da internet.
[10] Protocolo de segurança que protege as telecomunicações via internet para serviços como SMTP, HTTPS e outros tipos de transferências de dados.

O AMPQ funciona eficazmente em ambientes de múltiplos clientes fornecendo um meio para delegar as tarefas, e fazer com que os servidores lidem com solicitações imediatas mais rapidamente.

A figura 8 ilustra o fluxo de troca de mensagens entre o produtor e o consumidor usando o protocolo AMQP. Numa primeira etapa, o broker recebe a mensagem proveniente do produtor e coloca-a numa fila de espera. Por sua vez, o "binding" define a relação entre uma fila de mensagens e uma troca, e fornece os critérios de roteamento de mensagens. Posteriormente a mensagem é enviada para o consumidor.

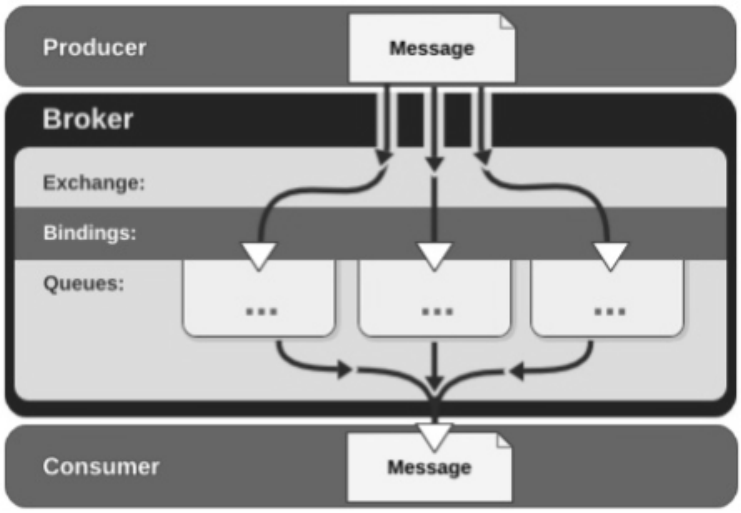


Figura 8 – Fluxo de troca de mensagens utilizando o protocolo AMQP.

2.4Plataformas IoT

Existem essencialmente quatro tipos de plataformas, as *end-to-end*, as de gestão de conectividade, as de cloud IoT e as plataformas de dados (Lee, 14). Fundamentalmente, as plataformas *end-to-end* fornecem as ferramentas de hardware, software, conectividade, segurança e gestão de dispositivos, de modo a lidarem com milhões de conexões em simultâneo. Fornecem também todas as funcionalidades que são necessárias - atualizações de firmware OTA[11], gestão de dispositivos, conexão à *cloud*, *cellular modem*, etc - para

[11] Métodos de distribuição de novas atualizações de software

conectar e monitorizar todos os dispositivos online. Um exemplo deste tipo de plataforma é a Amplify desenvolvida pela Fujitsu, que é abordada e analisada nesta dissertação. Outro exemplo é a plataforma Reply Smart Home[12] e a empresa de tecnologia Mainflux Labs[13].

Por outro lado, as plataformas de gestão de conectividade oferecem soluções de gestão de conexão de baixo consumo e custo através de tecnologias Wi-Fi e móveis.

As plataformas de cloud visam libertar o utilizador da complexidade de criar a sua própria pilha de rede complexa e oferecem os serviços de back-end (além de outros serviços) para monitorizar e lidar com milhões de conexões simultâneas de dispositivos. Um exemplo é a plataforma AWS IoT.

Naturalmente, todas as plataformas IoT gerem dados de alguma forma. Mas as plataformas de dados de IoT combinam muitas das ferramentas que são necessárias para gerir e analisar os dados dos dispositivos, nomeadamente através de visualização. Um exemplo é o projeto Kaa abordado na secção seguinte (2.4.1).

2.4.1 Projeto Kaa

Kaa é uma plataforma de *middleware* direcionada para IoT. Permite implementar soluções IoT completas, aplicações conectadas e produtos inteligentes. Oferece um conjunto de recursos IoT de nível empresarial, que podem ser facilmente utilizados para implementar um leque alargado dos casos de uso de IoT, fornecendo uma estrutura clara dos recursos e extensões de IoT para diferentes tipos de aplicações, como por exemplo, para a agricultura, indústria, desporto e *fitness*, telecomunicações, setor automóvel, setor da saúde, *smart cities* e *smart energies* (figura 9).

Em relação ao *hardware* a plataforma é agnóstica, ou seja, é compatível com praticamente qualquer tipo de dispositivos, sensores e portas TCP. Permite gerir um número ilimitado de dispositivos conectados, configurar a interoperabilidade entre os mesmos, colecionar e analisar dados de sensores, monitorizar a performance dos dispositivos em tempo real, distribuir atualizações de *firmware* e criar serviços em *cloud* para produtos inteligentes. Suporta *hardware* como por exemplo Edison, Beaglebone, Raspberry Pi, LeafLabs, Texas Instruments CC3200, ESP8266, Intel Edison Compute Module, Samsung Artik

[12] https://www.reply.com/br/topics/internet-of-things/smart-home
[13] https://www.mainflux.com/

5, Udoo. No que diz respeito ao armazenamento e processamento distribuído de dados, o Kaa suporta as ferramentas Hadoop, mongoDB, Oracle, Cassandra, Spark, Couchbase, CDAP, Kafka, entre outros. Em relação aos sistemas operativos, suporta Android, iOS, Linux, Snappy Ubuntu, QNX e Windows. Isto reflete uma grande abrangência por parte da plataforma Kaa, sendo este um dos aspetos importantes numa plataforma IoT.

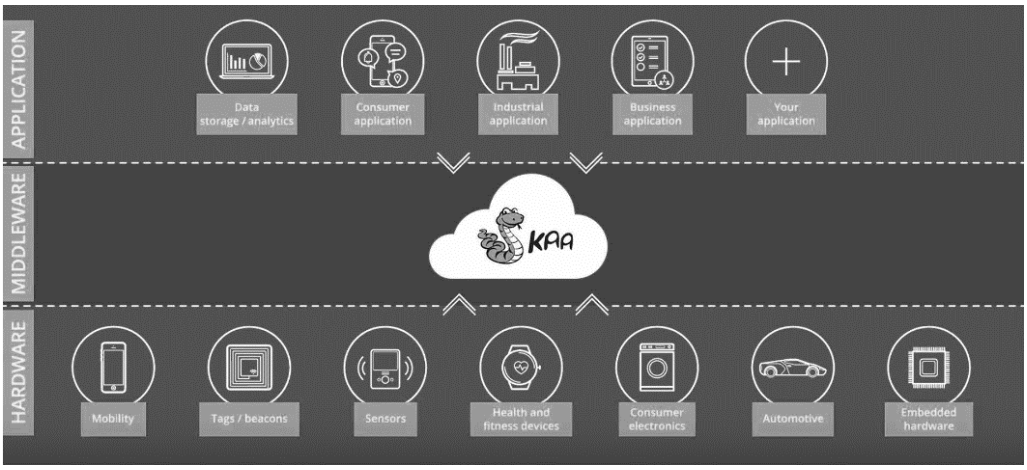


Figura 9 – Esquemático das aplicações e do hardware suportado pelo middleware do Kaa (Retirada de (CyberVision, 2018)).

A plataforma Kaa consiste em três componentes principais: o **servidor Kaa**, **extensões Kaa**, e **SDKs**. O servidor Kaa é a parte de back-end, utilizado para gerir clientes, aplicações, utilizadores e dispositivos, expondo interfaces de integração e oferecendo recursos administrativos. As extensões Kaa são módulos de software independentes, que adicionam funcionalidades à plataforma. Os SDK são bibliotecas (C, C++, Objective-C, Java) que fornecem APIs do lado do cliente para vários recursos da plataforma e lidam com a comunicação, o armazenamento de dados, a persistência, etc. Servem para facilitar a criação de aplicações cliente que podem ser executadas em diferentes dispositivos e que funcionam em conjunto com o cluster Kaa. No entanto, aplicações de clientes que não usam o SDK Kaa também são possíveis. A tabela seguinte mostra as plataformas suportadas por cada tipo de SDK.

Tabela 1 – Plataformas suportadas por diferentes tipos de SDKs. *(CyberVision, 2018)*

Plataforma	C	C++	Objective-C	Java
Linux	X	X	-	X
Windows	-	X	-	X
QNX Neutrino RTOS	X	-	-	-
Generic Desktop	-	-	-	X

Android	-	-	-	X
iOS	-	-	X	-
Raspberry Pi	X	X	-	-
Intel Edison	-	X	-	-
Beaglebone	X	X	-	-
Samsung Artik 5	-	X	-	-
UDOO	X	-	-	-
Texas Instruments CC3200	X	-	-	-
ESP8266	X	-	-	-

Os nós do servidor Kaa utilizam Apache ZooKeeper[14] de modo a coordenarem os serviços. Os nós interligados compõem um cluster Kaa associado a uma instância particular Kaa, e requerem instâncias de BD NoSQL e SQL para armazenar os dados e metadados dos nós de extremidade, tal como se demonstra na figura 10. Os nós Kaa num cluster executam uma combinação de serviços de controlo, operações e *bootstrap*[15].

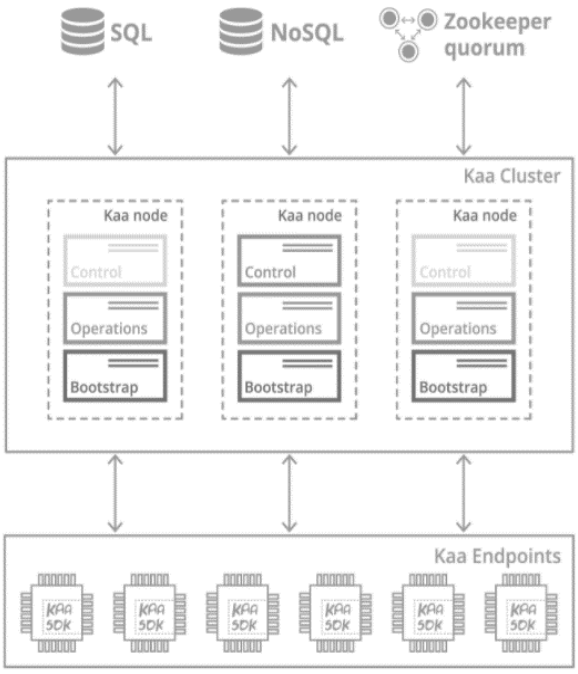


Figura 10 – Arquitetura da plataforma IoT Kaa (Retirada de (CyberVision, 2018)).

[14] Serviço centralizado que mantém informações de configuração, nomeação, fornece sincronização distribuída e serviços de grupo.

[15] Não se trata da conhecida *framework* para HTML, CSS e JavaScript, mas sim de um dos três principais tipos de serviços utilizados em Kaa.

O serviço de controlo faz uma gestão dos dados gerais do sistema, processa chamadas da API UI da web e de sistemas integrados externos, e envia notificações aos serviços de operações. Este serviço mantém uma lista atualizada dos serviços de operações disponíveis, recebendo continuamente essas informações do ZooKeeper. Além disso, o serviço de controlo fornece um componente administrativo, que utiliza as APIs do serviço para fornecer uma interface para gerir administradores, contas de utilizadores, aplicações e os respetivos dados, etc. Para fornecer Highly-Available, um cluster Kaa deve incluir pelo menos dois nós com o serviço de controlo habilitado. No modo HA, um dos serviços de controlo atua como ativo e o(s) outro(s) funciona(m) no modo de espera. No caso da falha do serviço de controlo ativo, o ZooKeeper notifica um dos serviços de controlo à espera e promove-o a serviço de controlo ativo.

É possível configurar um cluster Kaa com o serviço de operações habilitado para cada nó, funcionando todas as instâncias em simultâneo. Assim caso ocorra a interrupção de um serviço de operações, os pontos finais adaptam-se alterando automaticamente para outros serviços de operações disponíveis. Deste modo o servidor Kaa tem a capacidade de equilibrar a carga em tempo de execução.

Em suma, a plataforma Kaa permite a gestão de dados dos objetos conectados e das infraestruturas de *back-end*, fornecendo o servidor e os componentes SDK para os nós de extremidade. Os SDK são incorporados no dispositivo conectado, implementando a troca de dados bidirecional em tempo real com o servidor. Esta plataforma apresenta características como a flexibilidade, o facto de ser multiusos e ser 100% *open source*.

2.4.2 Ubiquitousware – Fujitsu

A solução IoT Ubiquitousware, desenvolvida pela Fujitsu, tem um conjunto de soluções desenhadas para converter dados de sensores em inputs importantes.

A solução disponibilizada permite o seu uso imediato no local, sendo fornecida através da cloud ou por integração parcial nos produtos do cliente, dependendo das suas necessidades. A solução é composta por três componentes principais, os sensores que coletam os dados, os algoritmos que ajudam a analisar dados e o software que confirma os eventos detetados. Combinando isto, a Fujitsu disponibiliza cinco soluções por áreas de

aplicação específicas: ***Driver Safety, Worker Safety, Location Monitoring, Worker Efficiency*** e ***Intelligent Care.***

A solução **Driver Safety** (figura 11) baseia-se num sensor *wearable* que deteta a sonolência e o cansaço com base na informação do ritmo cardíaco. Se o sensor detetar o aparecimento de sinais de sonolência, notifica o condutor e o seu supervisor. O dispositivo inclui ainda um algoritmo de aprendizagem e calibração cuja precisão aumenta com a utilização. Os condutores podem aceder a uma aplicação através de um smartphone e verificar o seu estado atual ou o registo do dia para identificarem momentos de maior sonolência. A aplicação permite ainda analisar que percursos estão a aumentar a sonolência e planear alterações no percurso para evitar o cansaço causado pela repetição.

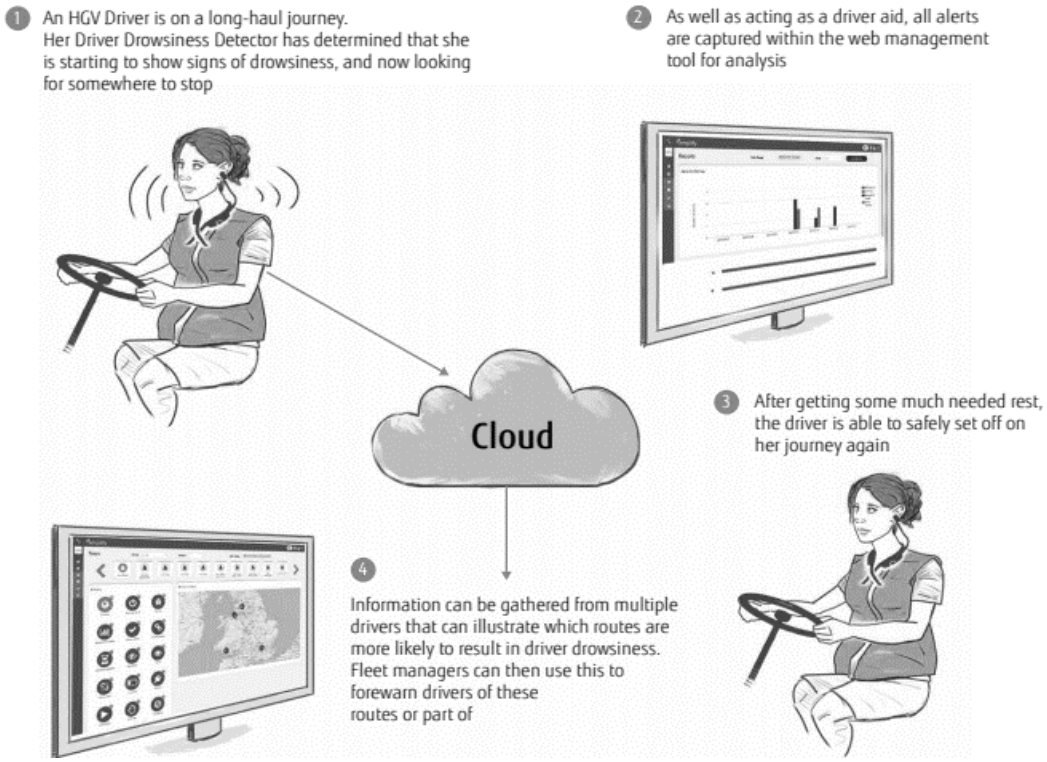


Figura 11 – Exemplo do fluxo de utilização da solução Driver Safety.

Gerir a segurança dos trabalhadores é fundamental, especialmente quando estes trabalham em situações de risco, como por exemplo, o trabalho em suspensão. Para estes cenários existe a solução **Worker Safety** (figura 12), que tem como funcionalidade prever problemas como o stress por calor. É possível, por exemplo, detetar quedas com precisão através de dados de aceleração e pressão barométrica, e enviar uma equipa de auxílio para a localização identificada. O mesmo princípio acontece com a solução **Location Monitoring** que permite gerir a localização de equipas e de ativos no terreno assegurando o envio de uma resposta para a localização certa se ocorrer um incidente.

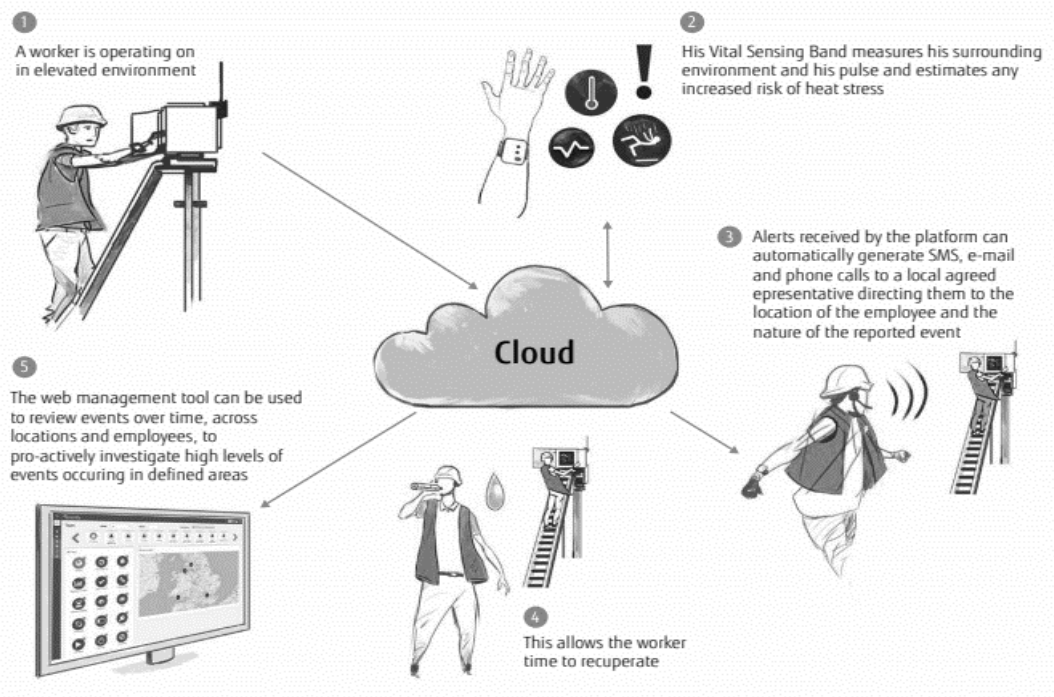


Figura 12 - Exemplo do fluxo de utilização da solução Worker Safety

O **Head Mounted Display**, da solução **Worker Efficiency** (figura 13) permite que os engenheiros comuniquem remotamente com outros colegas que se encontram no local da assistência, e partilhem em tempo real o que estão a ver. Esta solução permite acelerar a transferência de conhecimento e experiência aumentando a eficiência.

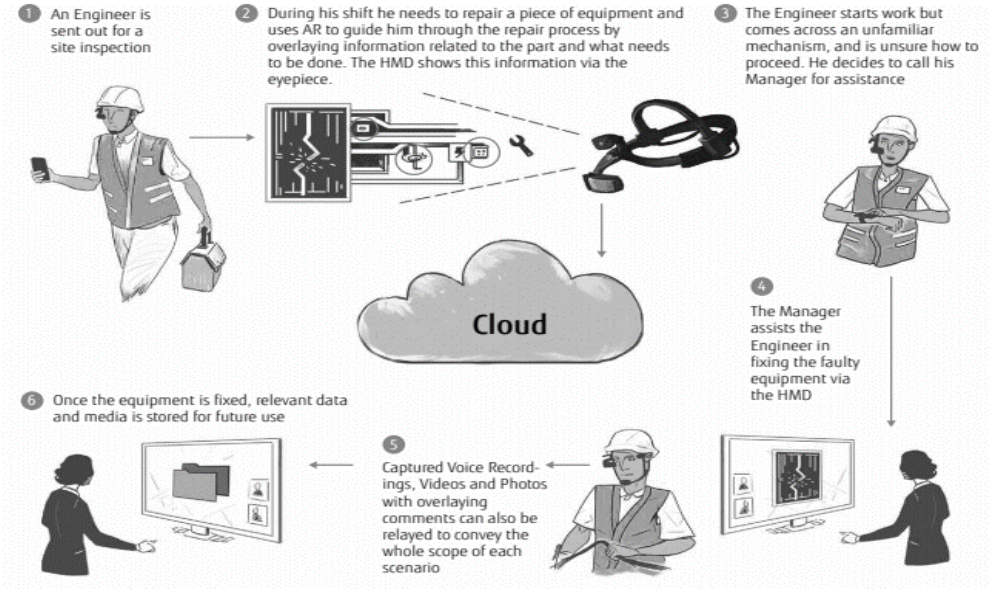


Figura 13 - Exemplo do fluxo de utilização da solução Worker Efficiency.

A solução ***Intelligent Care*** é destinada a serviços e cuidados de assistência domiciliária e em residências assistidas, assegurando a privacidade e protegendo a informação da pessoa. A estação de monitorização remota (**Remote Station**) reconhece incidentes através de análise de som, nomeadamente análise do discurso, tosse e respiração. Auxilia também na monitorização dos fatores externos como a temperatura e humidade. O sistema reconhece “eventos” através da análise de som, mantendo assim a privacidade. O sistema deteta anomalias comportamentais e de saúde, analisando dados de vários sensores. Os dados de um microfone ajudam a determinar a frequência e a intensidade da tosse e do ronco e a presença de ruído extraordinário, etc. Os dados dos sensores de presença ajudam a determinar a localização dos residentes e os sensores de temperatura e umidade ajudam a determinar as condições da sala que podem afetar a saúde. Em situações de emergência, os hóspedes podem pressionar o botão "emergência" ou "consulta" para falar com alguém no c*all center*. A equipe do c*all center* também pode falar através do equipamento sempre que forem detetadas anomalias comportamentais para verificar o estado do residente. Esta solução inteligente promove a autonomia e um estilo de vida mais saudável. Na figura 14 encontra-se descrito um fluxo de uma situação em que é detetada uma tosse anormal, ou seja, fora dos padrões detetados para aquele utilizador.

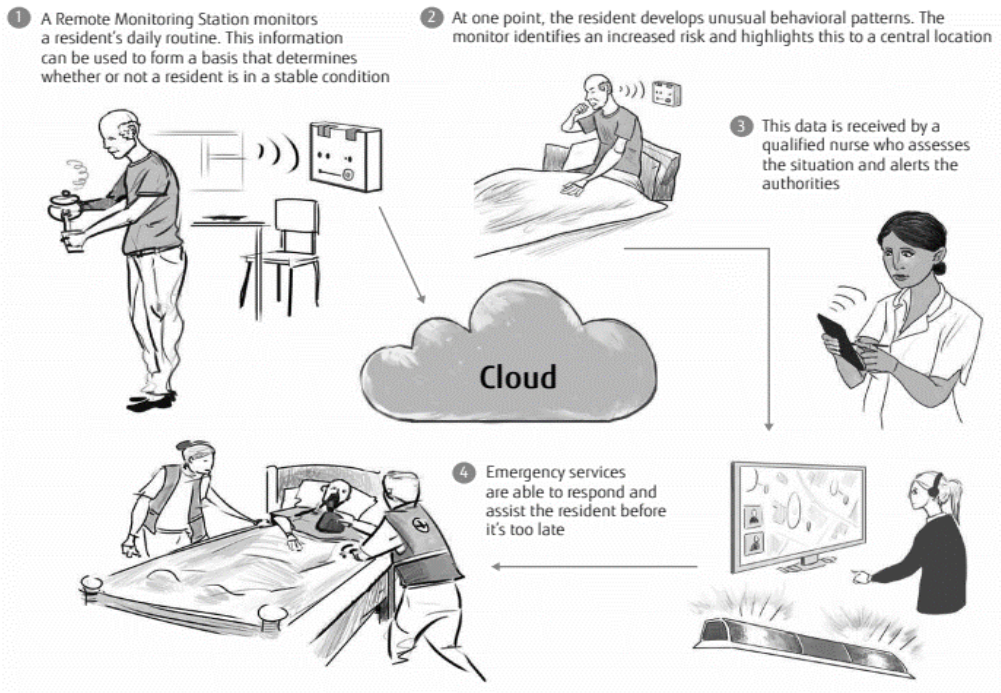


Figura 14 - Exemplo do fluxo de utilização da solução Intelligent Care.

2.4.3 Amazon (AWS) IoT

Os serviços IoT que a Amazon IoT oferece são, o **AWS Greengrass** e o **AWS IoT**. O último subdivide-se em seis serviços, nomeadamente **AWS IoT Core**, **AWS IoT 1-Click**, **AWS IoT Analytics**, **AWS IoT Button**, **AWS IoT Device Defender** e **AWS IoT Device Management**.

A **AWS IoT** é uma plataforma de construção de aplicações IoT que utiliza a *cloud* pública da AWS para armazenar, processar e analisar os dados dos dispositivos.

O **AWS Greengrass** é um software que permite executar com segurança recursos locais dos dispositivos, sistemas de mensagens, armazenamento de dados em cache, sincronização e inferência de aprendizagem de máquinas para dispositivos conectados. Com o **AWS Greengrass**, os dispositivos conectados podem executar funções do AWS Lambda, manter o sincronismo de dados de dispositivos e comunicar com outros dispositivos de forma segura, mesmo quando não estão conectados à Internet. O AWS Lambda é usado para garantir que os dispositivos respondem rapidamente a eventos locais.

O sistema Amazon FreeRTOS é um sistema operativo *open source*, baseado no Kernel FreeRTOS. Destina-se a microcontroladores e pretende facilitar a programação, implantação, armazenamento de dados em cache e sincronização.

O **AWS IoT Core** oferece uma gestão da plataforma em *cloud*, permitindo que dispositivos conectados interajam com facilidade e segurança com aplicações em *cloud* e outros dispositivos. Mesmo quando não se encontram conectados é possível acompanhar e comunicar com todos os dispositivos, independentemente de usarem protocolos de comunicação diferentes, tal como HTTP, WebSockets e MQTT, sendo estes os protocolos de comunicação que o **AWS IoT Core** suporta.

O serviço **AWS IoT 1-Click** torna mais fácil ativar funções **AWS Lambda** em dispositivos que executam uma ação específica. Tendo como exemplo prático a situação em que numa loja de serviço ao cliente, o mesmo tem a possibilidade de avaliar a sua experiência geral numa escala de um a cinco, podendo para tal pressionar um dos cinco botões disponíveis.

O **AWS IoT Button** é um dispositivo programável e de fácil configuração, baseado no **Amazon Dash Button**. Trata-se de um produto projetado para *developers* que pretendem dar os primeiros passos no **AWS IoT Core**, **AWS Lambda**, **Amazon DynamoDB** e **Amazon SNS**, sem que seja necessário escrever código específico para cada dispositivo. É possível ajustar o botão na *cloud*, de modo a configurar os cliques dos botões com a possibilidade de contar ou rastrear itens, ligar ou alertar alguém, iniciar ou parar algum serviço, solicitar serviços ou fornecer feedback. Este dispositivo tem um custo de vinte dólares e está disponível em duas versões, Wi-Fi e rede telefónica, contudo não está disponível em Portugal. Um exemplo prático de um caso de uso deste dispositivo é a sua colocação na zona do chuveiro, que após ser ativado solicita um pedido de ajuda e posteriormente essa notificação é enviada a um familiar. A figura 15 ilustra o fluxo de informação desde o momento que o utilizador pressiona o botão até ao envio de uma mensagem.

Para lidar com a enorme quantidade de dados que se geram, foi criado o serviço **AWS IoT Analytic,** que facilita a execução de análises sofisticadas em volumes maciços de dados IoT, sem haver preocupação com o custo e a complexidade necessários para a desenvolvimento de uma plataforma de análise IoT própria.

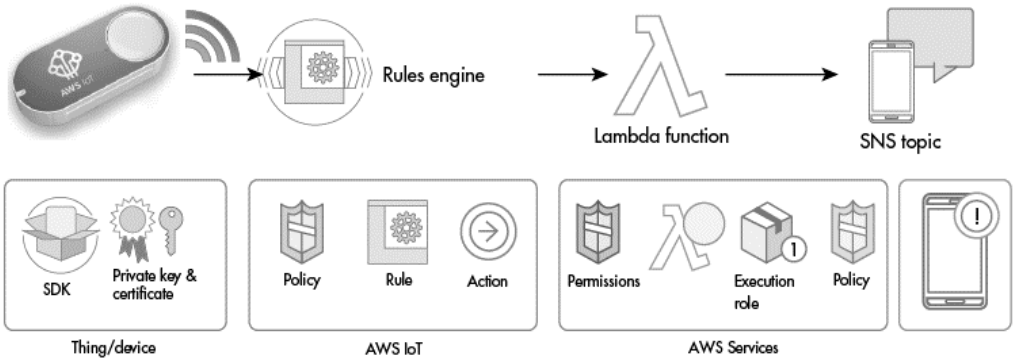


Figura 15 – Fluxo de informação da utilização do AWS IoT Button (Retirada de AWS Webinar).

Para gerir remotamente os dispositivos IoT, a Amazon disponibiliza o serviço **AWS IoT Device Management** que pretende integrar, organizar, monitorizar e gerir remotamente dispositivos IoT em grande escala.

O **AWS IoT Device Defender** é um serviço que ajuda a proteger a frota de dispositivos IoT. Este serviço está continuamente a realizar uma auditoria às políticas de segurança associadas aos dispositivos, para garantir que não estão a divergir das melhores práticas de segurança. Permitindo assim monitorar dispositivos de forma a serem detetados comportamentos divergentes dos comportamentos adequados.

2.4.4 MpDS (Medical Pre-Diagnostic System)

O projeto MpDS surgiu de uma parceria entre a F3M – Informations Systemas S.A e a Fraunhofer Portugal. Este projeto tem como objetivo a investigação e desenvolvimento de uma solução mobile que permita a aquisição de imagens médicas, com diferentes tipos de magnificações. Pretende-se que esta solução seja intuitiva e de fácil utilização, de modo a ser utilizada por parte do staff clínico especializado e não especializado. A solução mobile irá dar resposta a necessidades de recolha e processamento de imagens médicas, de modo a apoiar o diagnóstico clínico em diferentes contextos, nomeadamente a análise de sinais cutâneos, úlceras e amostras de sangue.

O projeto visa assim atingir três grandes objetivos:

- Simplificar o processo de recolha e armazenamento de imagens médicas (macroscópicas, dermoscópicas e microscópicas);

- Utilizar algoritmos de processamento de imagem na *cloud* para análise automática de imagens de sinais cutâneos;
- Utilizar algoritmos de processamento de imagem localmente no smartphone, com o objetivo de dar suporte em tempo-real à aquisição e validação da focagem das imagens adquiridas.

Como resultado do projeto, os autores esperam obter uma plataforma de apoio ao diagnóstico clínico, capacitando as organizações/unidades de saúde com uma nova ferramenta de apoio aos profissionais de saúde na tomada de decisão, que se traduzirá em ganhos de operação bastante importantes nesta área. Espera-se que o projeto tenha um elevado impacto no contexto global, permitindo a:

- Otimização de recursos em unidades locais de saúde;
- Redução do número de consultas presenciais;
- Redução do número de casos clínicos retornados pelos departamentos de especialidade.

O projeto MpDS é composto por: (i) uma aplicação Android denominada MpDS; (ii) uma aplicação multiplataforma, ou seja, iOS e Android, designada *Second Opinion* MpDS; (iii) um gestor de conteúdos; e; (iv) o software de gestão de unidades de saúde. O primeiro componente, permite ao utilizador visualizar a informação do paciente assim como abrir um novo caso clínico de sinais cutâneos, úlceras e/ou amostras de sangue tendo como base a recolha de imagens fotográficas. A aplicação *Second Opinion* MpDS, tem como principal função permitir a "discussão" entre utilizadores sobre um determinado caso clínico, tendo como base fotografias identificativas do caso e informação essencial do paciente, protegendo a sua privacidade e os seus dados. O gestor de conteúdos, permite ao cliente visualizar um conjunto de dados estatísticos tais como, número de casos clínicos em aberto, número de imagens fotográficas captadas, etc. Por último, o software de gestão de unidades de saúde permite às instituições/clinicas efetuar todo o processo de gestão do paciente, assim como a gestão de stocks, gestão da área financeiro e do *staff* clínico, etc.

O desenvolvimento de um algoritmo capaz de identificar e fotografar automaticamente tanto os sinais cutâneos como as feridas fica a cargo do instituto Fraunhofer Portugal. É de salientar que apenas o algoritmo de análise de sinais cutâneos auxilia o profissional de saúde, dando-lhe um feedback do nível de risco do mesmo. As

restantes imagens obtidas nos diferentes módulos, requerem a análise de um profissional qualificado na área.

2.4.5 Resumo

A tabela 2, sintetiza a informação analisada nas quatro plataformas abordadas (Kaa Project, AWS IoT, Ubiquitousware e MpDS), tendo em consideração algumas características relevantes. É de salientar que o símbolo "X", apresentando na tabela 2, representa que a informação não foi encontrada ou devidamente esclarecida.

Tabela 2 – Informação sintetizada das plataformas IoT abordadas.

Características	***Kaa Project***	***AWS IoT***	***Ubiquitousware***	***MpDS***
É um serviço grátis?	Sim	Não	Não	Não
É um sistema *OpenSource*?	Sim (Carece de Licença Apache 2.0)	Não	Não	Não
Que tipo de hardware suporta?	Agnóstico de hardware	Smartphones e tablets Android, iPhones e ipads	Smartphone	Smartphones e tablets Android, iPhones e ipads
Sistema(s) operativo(s) que suporta	Linux, Windows	Android (aplicações móveis), Windows	Android	Android, iOS
Linguagem(ens) de programação em que foi implementada	C/C++, Java, Objective-C	X	X	Kotlin, C#, Angular7
Linguagem(ens) de programação suportadas	C/C++, Java	Java, JavaScript, C#, .NET, PHP, Python, Ruby, GO, C++.	X	X
Quais são os protocolos de comunicação?	MQTT, CoAP	HTTP, MQTT	X	HTTP
Quais são as tecnologias de comunicação?	X	Bluetooth	Bluetooth, Wi-fi	Wi-fi
Padrão de troca de mensagens	X	Publisher/Subscriber	X	X
Formatos de dados	JSON	JSON	X	JSON
Cloud	Integrável em qualquer serviço de cloud	AWS IoT Core	Baseada em SaaS; Fujitsu Cloud Service IoT Plataform	Azure Cloud Services

2.5Relações entre Arquiteturas de referência WSO2, Protocolos e Plataformas

A arquitetura de referência (AR) WSO2, abordada na subsecção 2.2.1, apresenta seis camadas principais: (i) **Dispositivos**, (ii) **Comunicações**, (iii) **Agregação/Barramento**, (iv) **Processamento de Eventos e Análise**, (v) **Gestão de Dispositivos e Identidade**, (vi) **Gestão de Identidade e Acesso**.

Analisando a relação do projeto MpDS com AR WSO2 em termos da camada de **Dispositivos**, observa-se que estes são smartphones e tablets Android, iPhones e iPads, estes possuem hardware de comunicação incorporado, concretamente Wi-Fi. É ainda recomendado pelo WSO2 que o dispositivo possua um identificador único universal (UUID), que neste caso é global pois cada dispositivo móvel possui um identificador único.

Em termos da camada de **Comunicações**, a AR propõe o uso de três protocolos: MQTT, HTTP e o seu sobre o estilo arquitetural REST e CoAP. No projeto MpDS é utilizada uma abordagem REST em HTTP de modo a que as aplicações que suportam o sistema MpDS comuniquem com a API MpDS (figura 18), contudo não são implementados os protocolos MQTT e CoAP. A camada de Agregação/Barramento assume três papéis importantes:

- Implementar um servidor HTTP ou um broker MQTT que permita a comunicação entre dispositivos; o MpDS utiliza um servidor HTTP para interligar todos os dispositivos;
- Agregar e combinar informações de diferentes dispositivos que com ele comuniquem, sendo uma das responsabilidades do servidor MpDS.
- Assumir o papel de gateway convertendo mensagens MQTT para pedidos HTTP, e vice-versa, servindo como meio de ligação entre dispositivos que executam diferentes protocolos. Esta situação não se verifica no projeto MpDS dado que assenta unicamente em HTTP.

As informações geradas pelos diferentes dispositivos são armazenadas em bases de dados do tipo relacional podendo ser recuperadas e processadas, agindo posteriormente sobre elas, sendo esta uma das responsabilidades da camada de **Processamento de Eventos e Análise**.

O servidor MpDS gera um bearer Token por cada utilizador que inicie sessão na aplicação MpDS ou Second Opinion MpDS em diferentes dispositivos, atuando como chave de segurança, ou seja, tem de atuar como servidor OAuth2, validando os bearer Tokens e realizar os controlos de acessos, sendo esta mais uma das funções da camada de **Agregação/Barramento**. O token OAuth2 Refresh/Bearer é também associado a cada utilizador que inicie sessão na aplicação MpDS ou Second Opinion MpDS, munindo-se assim de um identificador auxiliar.

Examinando agora a ligação das soluções Ubiquitousware com a AR WSO2, em termos da camada de **Dispositivos**, observa-se que todas as soluções têm um dispositivo físico sendo que, por exemplo, a VB tem um modo de comunicação indireta, ou seja, apenas por Bluetooth, enquanto que a RMS possui ambos os modos de comunicação (direta e indireta), pois contém no seu hardware um interface de comunicação WiFi e Bluetooth. Todas as soluções têm um endereço Bluetooth, possuindo assim um identificador único universal tal como é recomendado pelo WSO2. Todas as informações obtidas através dos sensores das soluções Ubiquitousware são guardadas e processadas, de modo a atuar sobre elas, sendo estes os objetivos da Camada de **Processamento de Eventos e Análise**.

Um dos exemplos das soluções AWS IoT que integra a Camada de **Dispositivos** da arquitetura de referência WSO2, é o AWS IoT Button. Sendo que este detém um modo de comunicação direta (WiFI) e indireta (rede telefónica), possuindo ainda um UUID. Em termos da Camada de **Comunicações**, as soluções AWS utilizam os protocolos MQTT e HTTP tal como a arquitetura de referência WSO2 recomenda. A solução AWS IoT Core permite que dispositivos conectados interajam, com facilidade e segurança com aplicações em *cloud* e outros dispositivos, sendo a capacidade de agrupar e combinar informações de diferentes dispositivos, encaminhando-os para um dispositivo, um dos papeis da Camada de **Agregação/Barramento**. A solução AWS IoT assume as funções da camada de **Processamento de Eventos e Análise** que é responsável por obter informações (eventos) da camada de agregação/barramento, processá-los e agir sobre eles. A camada **Gestão de Dispositivos** está representada pela solução AWS IoT Device Management que permite gerir remotamente os dispositivos.

O projeto Kaa é agnóstico de hardware o que significa que em termos da camada de **Dispositivos**, suporta qualquer tipo de dispositivo físico. Sendo que o modo de comunicação e o facto de o dispositivo possuir um UUID, deverá ser analisado conforme o dispositivo

selecionado. Este projeto utiliza MQTT e CoAP como protocolos de comunicação, sendo estes dois dos protocolos recomendados na camada de **Comunicação**. O papel assumido pela camada de **Processamento de Eventos e Análise** integra o facto de este projeto colecionar e analisar dados provenientes dos dispositivos.

3. ABORDAGEM PARA A IMPLEMENTAÇÃO

As abordagens para a implementação às soluções refletem-se sobre os dispositivos Vital Band (VB) e sobre a Remote Monitoring Station (RMS) da Fujitsu, abordadas na subsecção 2.3.2 deste documento, e ao projeto MpDS abordado na subsecção 2.5.4. As escolhas destas soluções recaíram sobre o facto de se pretender um produto com presença neste mercado das soluções IoT, como é o caso dos produtos da Fujitsu. Por outro lado, o projeto MpDS é um projeto novo que integra diferentes componentes e no qual se pretende que no futuro seja uma solução IoT.

3.1Soluções/Plataformas

a) *Vital Band* (VB)

A VB possui os seguintes recursos: estimativa do ritmo cardíaco, deteção de queda, deteção de tensão térmica, monitorização do nível de esforço, monitorização da atividade física, deteção dos níveis de temperatura e humidade, deteção de desgaste da bateria ajudando a otimizar o consumo da energia. A VB conecta-se à plataforma *Enterprise Wearables* da Fujitsu, denominada por Plataforma de Serviço *Amplify*, para onde transmite os dados para serem processados e analisados. Essa conexão é efetuada através da aplicação Android "Gateway Software" e a Amplify é um serviço SaaS integrado e escalável que tem como objetivo integrar os dispositivos.

O kit da VB inclui uma estação de carregamento, um cabo USB, um carregador EU ou UK, uma pulseira e a unidade de sensor. A unidade de sensor é composta por um botão que permite ligar e desligar a unidade de sensor, dois leds de notificação e um led de baterias, um sensor, um sensor de pulso, um sensor de pressão atmosférica e um sensor de temperatura e humidade, tal como se verifica na figura 16. A tabela 3 permite visualizar quais as ações a tomar, e qual a informação visual e sensitiva que o utilizador recebe sobre cada estado, assim como a sua respetiva descrição.

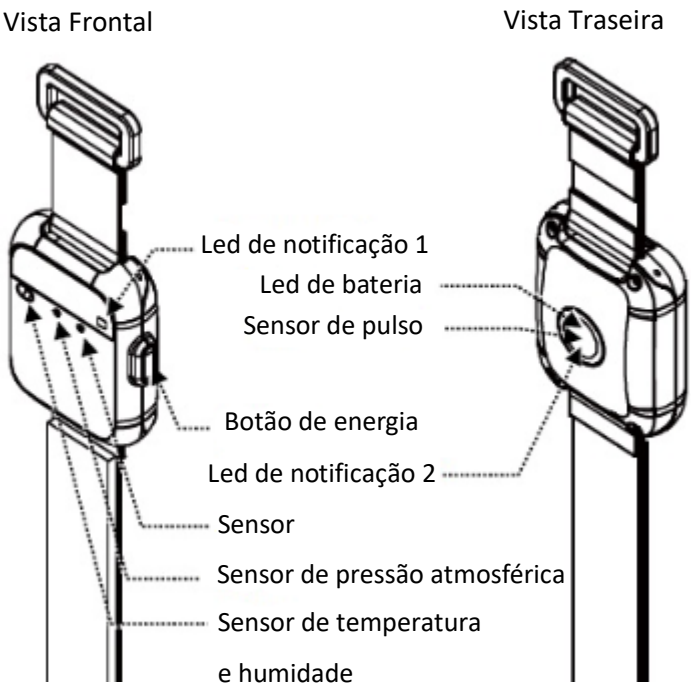


Figura 16 - Identificação dos componentes da Vital Sensing Band.

b) Remote Monitoring Station (RMS)

Tal como se descreve na subsecção 2.5.2 deste documento, a RMS, destina-se a serviços e cuidados de assistência domiciliária e em residências assistidas. A figura 17 ilustra os diferentes constituintes deste dispositivo, numa vista frontal, traseira e inferior, tais como o botão de cancelar a chamada, o led de energia, o altifalante, o botão de chamada de emergência, etc.

A RMS deverá ser colocada no espaço que o utilizador mais frequenta e o local deverá cumprir uma série de requisitos para que funcione corretamente, tal como se mostra na figura 18. Em relação à deteção de som, em qualquer tipo de espaço, o dispositivo deve estar a cerca de 80 a 130 cm do chão, e deve ser colocado no centro do espaço. Se por exemplo for colocado numa sala de estar, deverá respeitar a distância de cerca de 2 metros de fontes de ruído, como por exemplo janelas, aparelhos de ar condicionados, ventiladores, dispositivos com o som alto, etc, e o vento proveniente de qualquer aparelho não deverá atingir diretamente a RMS. Convém respeitar a distância máxima de 5 metros à RMS, contudo recomenda-se os 3 metros, e um ângulo de 30 graus. Caso seja instalada num quarto, a RMS deverá estar o mais próximo possível do alvo, aproximadamente 30 a 50 centímetros do travesseiro da cama.

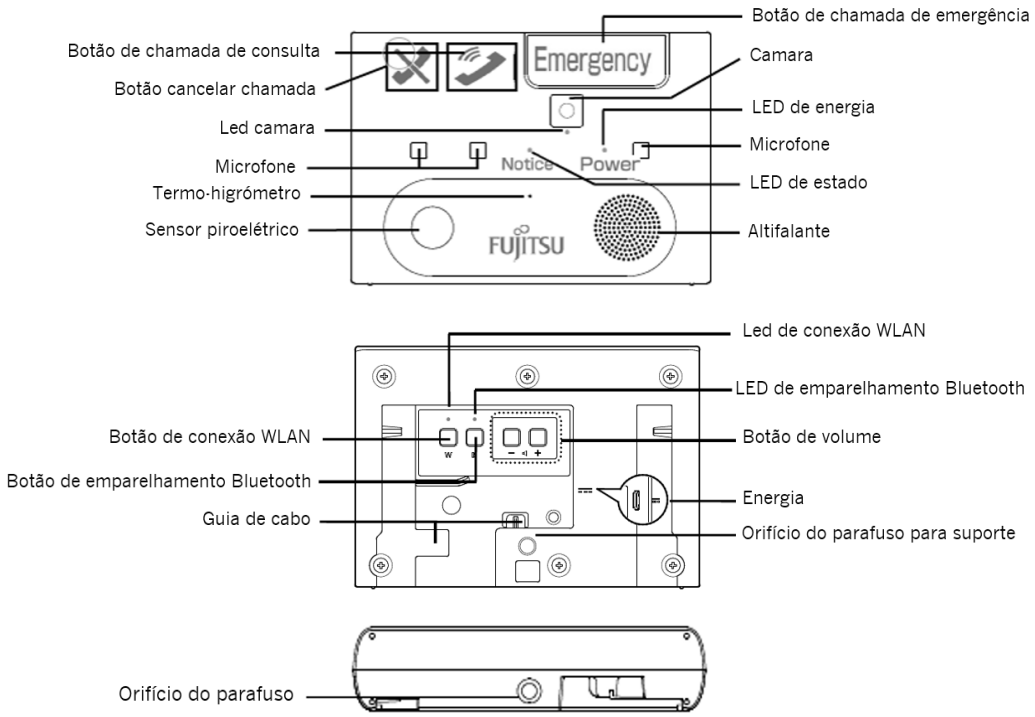


Figura 17 - Identificação dos constituintes da RMS em vista frontal, traseira e inferior.

O ângulo horizontal de deteção humana é de cerca de 102 graus, 90 graus no plano vertical e a distância até 8 metros. É de salientar que o dispositivo não é à prova de água, portanto não funciona corretamente em casas de banho. Em relação à câmara o ângulo de deteção é de 70 graus na horizontal e de 40 graus em na vertical. De notar que a RMS não consegue obter uma imagem clara se houver uma luz forte atingir a mesma ou houver luz de fundo.

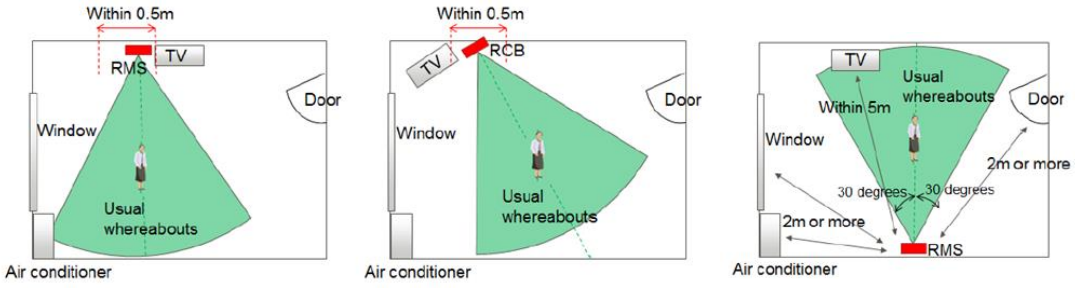


Figura 18 – Esquema dos requisitos a cumprir para a RMS.

c) Projeto MpDS

A figura 19 representa um esquema da API do projeto MpDS, onde é possível identificar quatros soluções que contemplam o sistema: (i) Gestor de Conteudos; (ii) MpDS; (iii) 2ª Opinião; e; (iv) Software de Gestão de Cuidados de saúde. O retângulo inferior exibe a base de dados SQL e os serviços *cloud* da Azure. Serviços esses que são: Azure Blob Storage e Azure Table Storage. O Azure Blob, que integra o Azure Storage, permite armazenar dados binários na cloud, nomeadamente qualquer tipo de arquivo acompanhado de metadados. É ainda possível controlar os acessos, alterando as permissões, ou deixando-os públicos. A API do Azure Storage segue a filosofia REST. O Table Storage é um serviço que armazena dados NoSQL estruturados na cloud, fornece um repositório chave-atributo sem schemas que permite adaptar mais facilmente os dados à medida que as necessidades evoluem.

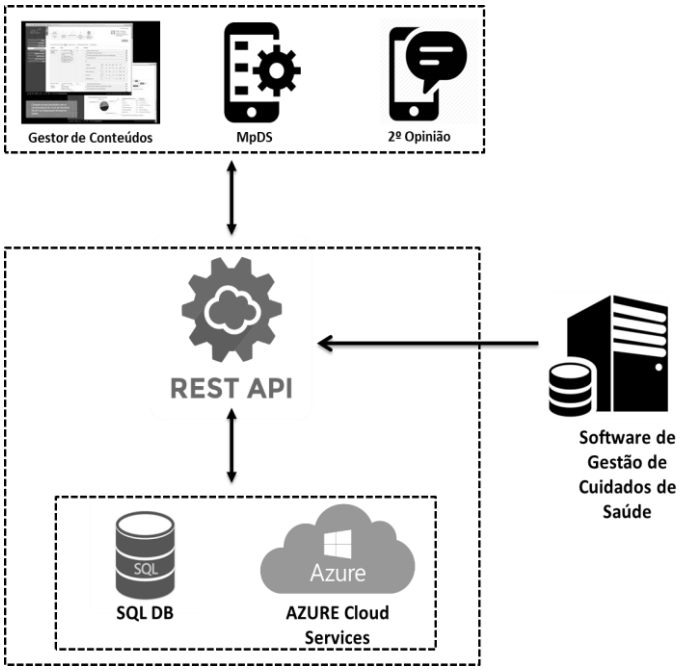


Figura 19 – Esquemático API projeto MpDS.

O Swagger é uma framework de software open-source que auxilia os programadores a projetar, criar, documentar e consumir serviços da Web Restful. Apesar de ser bastante reconhecido pela sua ferramenta Swagger UI, o conjunto de ferramentas Swagger inclui suporte para criação de documentação, criação de código e de casos de teste. O projeto Swagger API foi criado em 2011 por Tony Tam, cofundador técnico do site de dicionários

Wordnik. Esta framework foi utilizada, numa primeira fase, para testar a comunicação com a API MpDS.

3.2Plano

a) Vital Band (VB)

A configuração/programação abrange duas vertentes: a plataforma de serviço Amplify e a aplicação Android "Gateway Software", e o hardware VB. Primeiro é necessário fazer registo da VB e do smartphone na plataforma de serviço Amplify, tendo como base o endereço físico MAC do smartphone e da VB, e a instalação da aplicação Android num smartphone, é necessário configurar algumas definições na aplicação Android. essas Definições encontram se num ficheiro disponibilizado pela Fujitsu e que deve ser incorporado na aplicação Android de modo a preencher automaticamente os campos da configuração.

b) Remote Monitoring Station (RMS)

A configuração/programação engloba a aplicação Android "IoT Gateway Setting App", a RMS, um servidor e um cliente SIP. Após a instalação da aplicação Android, tal como acontece com a VB, é necessário incorporar um ficheiro, disponibilizado pela Fujitsu, com as respetivas configurações para a RMS.

c) MpDS

Apesar do projeto MpDS incluir vários componentes (aplicação MpDS, aplicação Second Opinion MpDS, gestor de conteúdos e software de gestão de cuidados de saúde) que comunicam com a API, a configuração/programação abrange apenas a comunicação entre a aplicação MpDS e a API, utilizando o Swagger UI como primeira abordagem de comunicação com a API MpDS.

Este projeto apresenta os seguintes requisitos funcionais:

1. O sistema deve permitir o acesso apenas a utilizadores registados na base de dados.

2. O sistema deve permitir escolher a instituição ao qual o utilizar deseja iniciar sessão caso tenha mais do que uma associada ao seu email.
3. O sistema deve permitir que o utilizador termine sessão.
4. O sistema deve permitir que o utilizador apenas tenha acesso à informação dos pacientes da instituição à qual iniciou sessão.
5. O sistema deve efetuar uma gestão de licenças por instituição.
6. O sistema deve possibilitar ao utilizador a captura de fotografias em modo manual e automático.
7. O sistema deve sincronizar a informação entre o software de gestão de cuidados de saúde e a API MpDS.
8. O sistema deve sincronizar a informação entre a API MpDS e a aplicação MpDS.
9. O sistema deve apresentar ao utilizador a possibilidade de efetuar a caracterização de uma.
10. Caso o utilizador preencha os campos de caracterização de uma ferida, devem ser apresentados os gráficos de evolução da mesma.
11. O utilizador deve poder alterar a sua imagem de perfil.

Em relação aos requisitos não funcionais são os seguintes:

1. O sistema deve possuir um design para mobile responsivo.
2. O sistema deve ter uma boa usabilidade.
3. O sistema deve cumprir as normas do RGPD.
4. A aplicação MpDS deve estar disponível para Android 8.0.0 e superior.

3.3Testes

Testar é um processo que tem como propósito encontrar falhas num sistema, tendo como objetivo a eliminação de erros ou a verificação de funcionalidades.

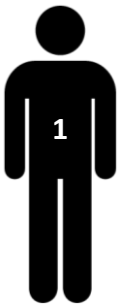
Os testes definidos classificam-se em três tipos: (i) Desempenho; (ii) Qualidade; e; (iii) Compatibilidade do Equipamento.

Nos testes de desempenho serão avaliados os comportamentos dos produtos, tendo em conta as funções para os quais foram projetados. Os testes de qualidade terão em consideração aspetos como a robustez dos produtos e a usabilidade dos mesmos. Por

último, os testes de compatibilidade dos equipamentos incidirão sobre a possibilidade de emparelhamento com diferentes gamas de dispositivos móveis, dentro da gama do Android, assim como a sua facilidade no modo de emparelhamento.

3.3.1 Vital Band (VB)

O dispositivo VB foi testado por três pessoas, tendo as mesmas as seguintes características:



- 25 anos
- 185 cm
- 73 Kg

- 54 anos
- 160 cm
- 68 Kg

- 25 anos
- 170 cm
- 69 Kg

A tabela 6 descreve o plano de testes realizado sobre a VB. Iniciando-se pela instalação e configuração da mesma, simulando posteriormente algumas ocorrências.

Tabela 3 – Descrição do plano de testes da Vital Band.

Utilizador	**Descrição**		**Duração**
Pessoa "N"	Instalação e Configuração		2 horas
	Simulação de ocorrências	Sofrer uma queda	-
		Saltar	2 minutos
		Dançar	10 minutos
		Corrida prolongada	5 minutos
		Várias corridas curtas	Intervalos de 30 segundos
		Temperaturas anormais	15 minutos

3.3.2 Remote Monitoring Station (RMS)

O dispositivo RMS foi testado em três divisões, sendo duas delas dois quartos e a terceira uma sala. Os dois quartos têm ambos 20 m^2 e a sala tem 48 m^2. A figura 20 esboça os desenhos das divisões nas quais foram efetuados os testes, especificando a localização da RMS e as respetivas distâncias às fontes de ruído (televisão, ar condicionados), portas e janelas.

A tabela 7 descreve o plano de testes realizado sobre a RMS. Iniciando-se pela instalação e configuração da mesma, simulando posteriormente algumas ocorrências.

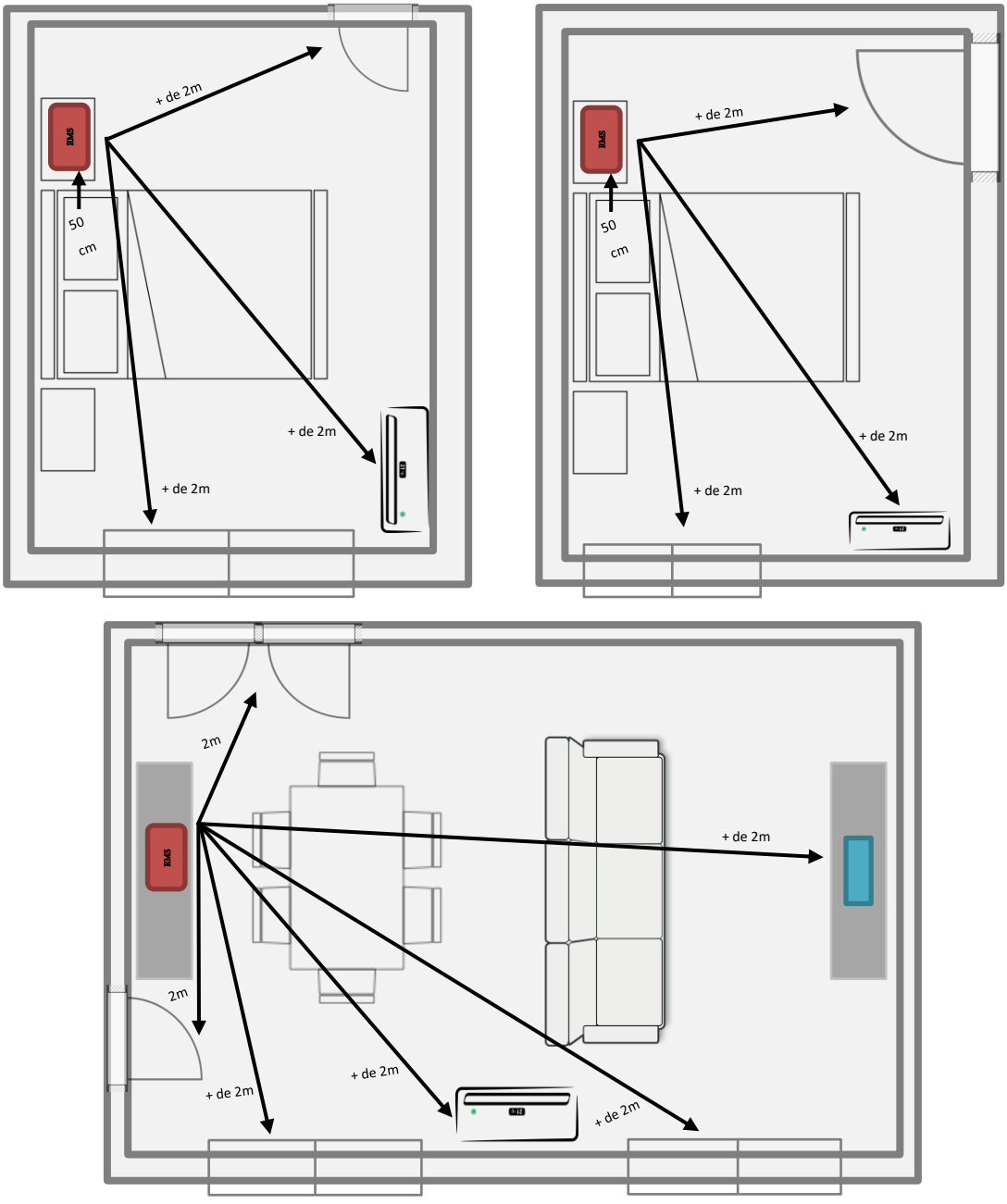


Figura 20 - (a) Divisão A; (b) Divisão B; (c) Divisão C.

Tabela 4 – Descrição do plano de testes da Remote Monitoring Station.

Espaço	Descrição		Duração
Divisão "N"	Instalação e Configuração		2 horas
	Aprendizagem da rotina		72 horas
	Simulação de ocorrências	Televisão desligada	1 hora
		Presença humana não detetada	1 hora
		Tosse continua	5 minutos
		Ressonar	20 minutos
		Deixar cair objetos com diferentes pesos, volumes e durezas	5 segundos
		Temperaturas anormais	15 minutos
		Humidades anormais	15 minutos
	Acionar botão SOS		-
	Acionar botão Helpdesk		-

3.3.3 MpDS

Os casos de uso da solução MpDS apresentados na tabela 8, serão testados por três intervenientes não pertencentes ao staff clínico e com as mesmas características que os testadores da solução VB (subsecção 3.3.1)

Tabela 5 – Descrição do plano de testes efetuados à solução MpDS.

Caso de Uso	Descrição
1º - Iniciar Sessão	1. A aplicação envia um email de boas vindas com a password provisória e link de instalação da aplicação através da play store.
	2. O utilizador faz download da aplicação a partir da play store.
	3. O utilizador inicia sessão na aplicação MpDS introduzindo o seu email e a password que recebeu através do email.
	4. O utilizador define nova password de acesso.
	5. A aplicação envia um email de confirmação de alteração da password.
	6. O utilizador inicia sessão na aplicação introduzindo o seu email e a nova password.
	7. O utilizador seleciona a instituição na qual pretende iniciar sessão, caso esteja registado em mais do que uma instituição.
2º - Abrir Novo Caso Clínico	8. O utilizador pesquisa um utente.
	9. O utilizador inicia um novo caso clínico clicando no ícone da camara.

	10. O utilizador tira uma fotografia.
	11. O utilizador adiciona mais duas fotografias.
	12. O utilizador elimina a última fotografia.
	13. O utilizador sinaliza a ferida no desenho do corpo humano.
	14. O utilizador descreve as observações verificadas.
	15. O utilizador sinaliza a ferida no desenho do corpo humano.
	16. O utilizador inicia o processo de caracterização da ferida preenchendo os campos.
	17. O utilizador submete o novo caso clínico.
	18. O utilizador seleciona a data da próxima intervenção.
	19. O utilizador visualiza a sinalização da ferida no corpo humano.
	20. O utilizador visualiza a caracterização da ferida.
3º - Pedir Segunda Opinião	21. O utilizador clica na opção “Segunda Opinião” no menu que se encontra no fim do ecrã.
	22. O início de sessão na aplicação da Second Opinion MpDS é efetuado automaticamente.
	23. O utilizador seleciona outro utilizador com o qual deseja comunicar.
	24. O utilizador atribui um nome à conversa.
	25. O utilizador inicia a comunicação através de chat com outro utilizador.
4º Relatório	26. O utilizador visualiza a informação simplificada do utente selecionado.
	27. O utilizador visualiza a informação detalhada do utente selecionado.
	28. O utilizador visualiza as fotografias obtidas ordenadas pela mais antiga até à mais recente, sendo que a primeira foto visualizada é a mais recente.
	29. O utilizador visualiza as fotografias obtidas numa dada intervenção em modo tela completa.
	30. O utilizador aplica um intervalo de datas para a filtragem dos gráficos.
	31. O utilizador efetua uma manipulação dos gráficos (zoom, ativar/desativar linhas).
	32. O utilizador visualiza a localização da ferida na imagem do corpo humano.
5º Adição Nova Intervenção	33. O utilizador seleciona o botão “Nova Foto” que permite adicionar uma nova intervenção a uma ferida em aberto.
	34. O utilizador visualiza a localização da ferida no corpo humano sem possibilidade de a alterar.
	35. O utilizador visualiza as observações efetuadas anteriormente àquela ferida.
	36. O utilizador avança para o processo de caracterização da

	ferida.
	37. O utilizador visualiza a caracterização efetuada anteriormente àquela ferida, com possibilidade de edição de todos os campos.
	38. O utilizador atualiza campos da caracterização da ferida.
	39. O utilizador submete a nova intervenção.
	40. O utilizador seleciona uma data para a próxima intervenção.
	41. O utilizador seleciona “OK” para enviar a data da próxima intervenção.
	42. O utilizador visualiza a sinalização da ferida no corpo humano.
	43. O utilizador visualiza a caracterização da ferida.
	44. O utilizador seleciona a opção “Lista de Utentes” no menu inferior.
6º Lista de Histórico de Intervenções	45. O utilizador visualiza o histórico de intervenções, ou seja, os utentes dos quais se abriu um novo caso clínico.
7º Menu Lateral	46. O utilizador seleciona o ícone no canto superior esquerdo.
	47. O utilizador altera a sua fotografia de perfil.
	48. O utilizador termina sessão.
8º Recuperar Password	49. O utilizador seleciona a opção “Esqueceu-se da password?”.
	50. O utilizador insere o seu email e a nova password.
	51. O utilizador recebe um email de confirmação de alteração de password, no qual deve selecionar o botão “confirmar” para efetivar a sua alteração.
9º Iniciar Sessão Login Automático	52. O utilizador introduz a sua password.
	53. O utilizador seleciona a instituição na qual deseja iniciar sessão.

4. INSTALAÇÃO E CONFIGURAÇÃO

O número de projetos desenvolvidos com base no conceito IoT têm vindo a aumentar, contudo não existia regulamentação sobre este tipo de projetos.

O ETSI, Instituto Europeu de Normas de Telecomunicações, é uma organização europeia de normalização, que tem como missão a produção de normas europeias na área das telecomunicações, desenvolvendo igualmente atividade de pré-normalização e normalização nas áreas das tecnologias da informação e da radiodifusão televisiva e sonora. Em fevereiro de 2019, a ETSI lançou um documento de especificação técnica[16] relativo à cibersegurança para os consumidores IoT. Resumidamente a ETSI apresenta seis requisitos essenciais:

- Capacidade de ligação por cabo (ou ótica) à internet (ou à rede corporativa);
- Capacidade de inibir a ligação sem fio;
- Capacidade de alterar as credenciais de acesso;
- Capacidade de comunicação cifrada;
- Capacidade de atualização de firmware por parte do fabricante;
- Arquitetura de rede adequada para IoT's.

Com esta especificação técnica, o que a ETSI pretende é garantir que os requisitos de segurança são integrados na conceção e no desenvolvimento, auxiliando nomeadamente o cumprimento do RGPD (Regulamentação Geral de Proteção de Dados).

4.1Vital Band (VB)

Numa primeira instância é essencial adicionar e configurar os dispositivos na plataforma, para tal é necessário fornecer o endereço físico (MAC), assim como indicar qual o tipo de equipamento, entre seis opções (figura 21): "HMD Device", "Vital Band", "Location Badge", "Location Badge (EPP)", "Gateway Device" e "Virtual Gateway". Na figura 21, é possível visualizar a adição de três dispositivos, sendo dois deles do tipo gateway –

[16] https://www.etsi.org/deliver/etsi_ts/103600_103699/103645/01.01.01_60/ts_103645v010101p.pdf

“DeviceRita” e “Redmi”. O terceiro dispositivo trata-se da Vital Band denominada por “raloureiro VB”.

De modo a proceder-se à instalação e configuração da aplicação Android, liga-se o smartphone por cabo USB ao PC, e copia-se o apk da aplicação para o armazenamento interno do mesmo. Após a instalação da aplicação Android no smartphone é necessário reiniciar o smartphone. Posteriormente acede-se à pasta Fujitsu/IoTGESetting e cria-se uma pasta denominada “Settings” para onde se copia o ficheiro de configuração (disponibilizado pela Fujitsu). Este ficheiro contém informação a ser utilizada no preenchimento de alguns campos da configuração.

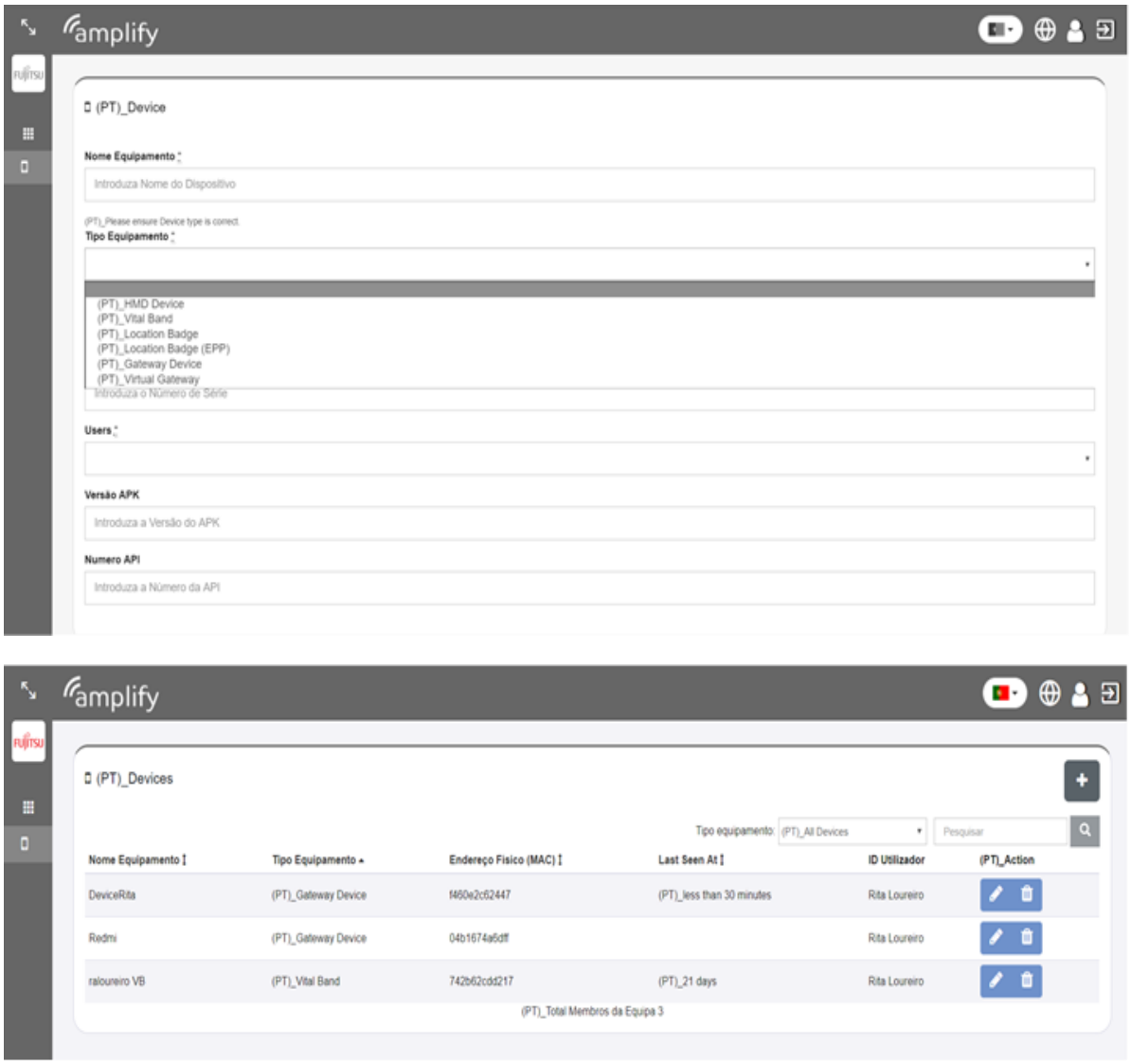


Figura 21 - Configuração de novos dispositivos na plataforma Amplify.

Depois de abrir a aplicação Android, toca-se nos três pontos que se encontram no canto superior direito para aceder à opção “Gateway Settings”. Na barra superior clica-se na opção “Get” e posteriormente em “Get setting values from settings file” de modo a importar as configurações do ficheiro. De seguida é necessário configurar alguns campos manualmente, nomeadamente “Device scan interval” e “Gps Measuring Cycle” e ativar o “Use GPS smartphone Data”. Para os dois primeiros escolheu-se o valor de 30 segundos para ambos. Por último, na barra superior clicar na opção “Update”, para que as configurações fiquem gravadas no servidor, e depois reiniciar novamente o smartphone. Este fluxo é mostrado através da figura 22.

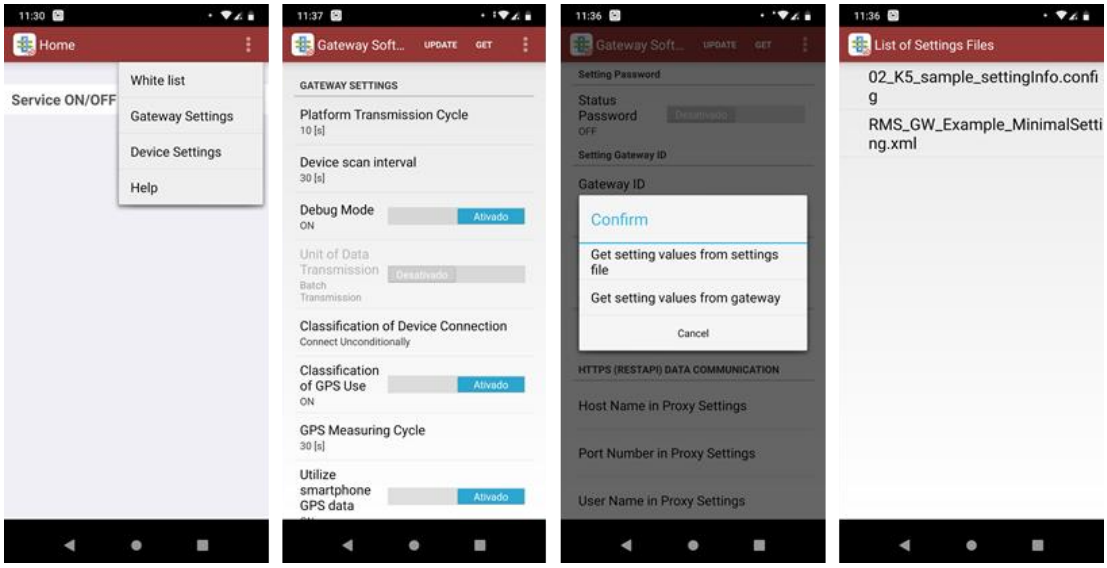


Figura 22 - Configuração da aplicação Android Gateway Software.

4.2Remote Monitoring Station (RMS)

A RMS, tal como a Vital Band, também tem uma aplicação Android de interface, denominada por “IoT Gateway Setting App”. Após ligar-se o smartphone por cabo USB ao PC e transferir-se a aplicação para o armazenamento interno, deve-se instalar a mesma. De seguida, a partir do PC, acede-se à pasta da aplicação, ao diretório Fujitsu/IoTGESetting e cria-se uma pasta que deverá ter o nome “Settings”, para onde deverá ser copiado o ficheiro de configuração (disponibilizado pela Fujistsu).

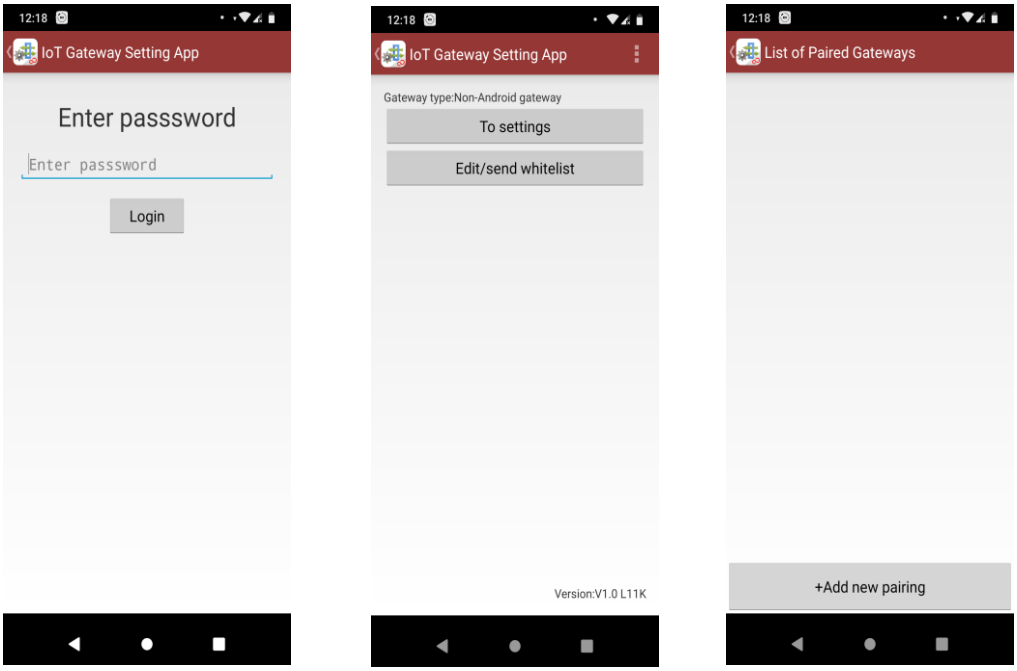


Figura 23 – (a) Layout de introdução de password; (b) Layout menu principal; (c) Layout da lista de dispositivos emparelhados

Ao executar pela primeira vez a aplicação deve-se introduzir uma password (figura 23 (a)), que poderá ser alterada na aplicação a partir dos três pontos que se encontram no canto superior direito.

Na parte de trás da RMS pressiona-se o botão "B" e quando o led começar a piscar significa que se encontra em modo de emparelhamento. É de ter em consideração que este modo só dura 30 segundos. De seguida seleciona-se o botão "To settings" (figura 23 (b)) na aplicação Android e seleciona-se "+Add new pairing" (figura 23 (c)). Surgirá então a lista de endereços de dispositivos Bluetooth de entre os quais deverá ser selecionado o endereço da RMS. Posteriormente, seleciona-se a opção "Remote Monitoring Station Settings" a partir da janela de alerta e escolhe-se o ficheiro que contém a informação de configuração da RMS.

No menu de configurações seleciona-se a opção "Configurações de informações de Wi-Fi" e definem-se os parâmetros necessários (tipo de segurança, SSID e password) para que a RMS se conecte à rede Wi-Fi local. Antes das informações serem enviadas para a RMS convém confirmar se todos os parâmetros se encontram de acordo com os seguintes:

SETTINGS FOR CONNECTION
|- Wi-Fi Information Settings [Configured above]
|- VoIP Information
Settings

|- NTP Server Information Settings [ntp.nict.jp + user = id + password = password]
|- Setting of Classification of Connection Destination for IoT Platform [FUJITSU Cloud Service IoT
Platform]
|- Setting of IoT Platform Data Communication Protocol [REST]

FUJITSU CLOUD SERVICE IOT PLATFORM
|- Broker Address Setting [api.sys2.iot.jp.fujitsu.com]
|- Port Number Setting [1883]
|- User ID Setting [INOIOT-010]
|- Password Setting[-----]
|- System Version Setting [v1]
|- Tenant ID Setting [INOIOT-010]
|- Base URI Settings for Connection Destination [http://api.sys2.iot.jp.fujitsu.com]
|- Connection Information
|- Sensor Table [iotSensor]
|- Device Table [iotDevice]
|- Event Table [iotSensorEvent]
|- Control Command REST [iotCommand]
|- Control Command / MQTT Request [iotCommandReq]
|- Control Command / MQTT Response [iotCommandRes]

SERVICER INFORMATION
|- Broker Address Setting [api.sys2.iot.jp.fujitsu.com]
|- Port Number Setting [1883]
|- User ID Setting [INOIOT-010]
|- Password Setting[-----]
|- Setting of Clould Notification Cycle [600 (s)]
|- MQTT Keep alive Setting [480 (s)]
|- Setting of Polling Interval for Control Commands [10 (s)]
|- Retry Count [3 (times)]
|- BLE Connection [Whitelist Mode]

SETTINGS FOR FUNCTION
|-
|- Service Status [ON]
|-

Os valores que se encontram dentro dos parênteses retos, são exemplos que podem variar de acordo com a implementação, estando sujeitos à plataforma e preferências do utilizador. Após verificação de todos os valores de configuração, volta-se ao menu principal e clica-se na opção “Restart”.

Finalizado o processo de instalação e configuração da aplicação Android é necessário configurar um servidor SIP e um cliente SIP de modo a que as chamadas de emergência e de consulta sejam realizadas, ou seja, que o cliente que utiliza a RMS consiga realizar uma chamada. Numa primeira instância é um fator importante compreender os conceitos associados ao servidor SIP e ao cliente SIP. Um servidor SIP é o componente principal de um IP PBX, e lida principalmente com a gestão das ligações SIP dentro da rede. Um cliente SIP é qualquer dispositivo de rede que envia solicitações SIP e recebe respostas SIP. O objetivo é estabelecer comunicações em tempo real (RTC), estabelecendo uma conexão com os servidores SIP. Como servidor SIP, a Fujitsu sugere o Brekeke, porém existem outros grátis e open source que devem ser considerados, tais como: Asterisk, Cipango, FreeSwitch, FreePBX, Issabel, Kamailio, OpenSIPS, SailFi, Yate, YXA, etc. O Ageet é o sugerido pela Fujitsu como cliente SIP, contudo o Zoiper, Bria, CSIPDSimple, Sipdroid, MizuDroid, etc, são outras das opções. Após análise, tantos dos servidores como dos clientes SIP, optou-se pelo Yate como servidor SIP e pelo MizuDroid como cliente SIP.

4.3MpDS

O Swagger UI, tal como descrito na subsecção 3.2 (c), permite visualizar e interagir com as APIs. Assim é possível testar um serviço que siga o formato OpenAPI, e onde conste um conjunto de ferramentas possíveis de efetuar, com os devidos parâmetros, o tipo de resposta e o endereço HTTP. O Swagger UI é um elemento relevante para o desenvolvimento deste projeto dado que o seu objetivo é permitir que a documentação evolua ao mesmo ritmo da evolução da API, por ser gerada automaticamente com base nas anotações do código. Permitindo aos programadores e testadores terem um entendimento exato do comportamento disponibilizado pelo serviço.

Dada a extensão dos métodos da API MpDS, só são considerados para efeitos de teste os métodos de início de sessão - “Login” - e de informações do paciente – “Patient” (figura 24).

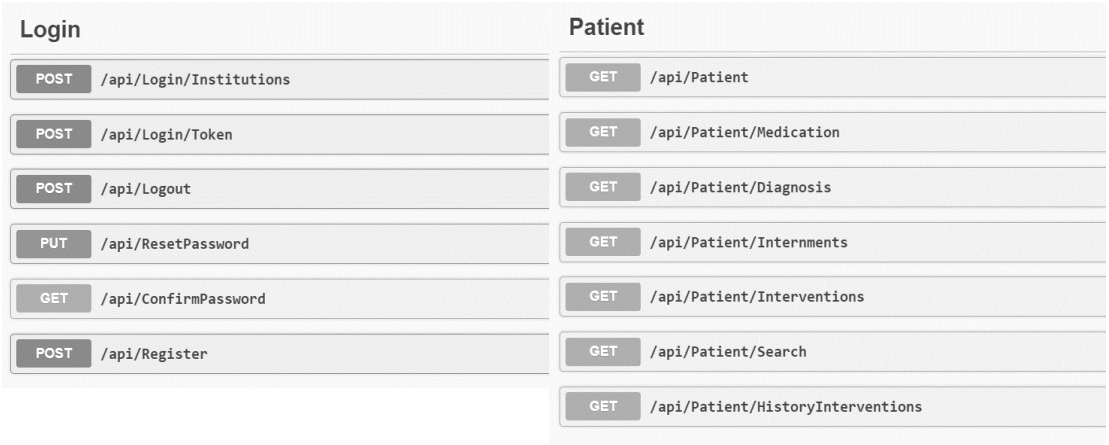


Figura 24 – Lista de pedidos que podem ser efetuados para obtenção/envio de informação do utilizador e dos pacientes.

O método PUT com a URI apresentada na figura 25, permite modificar a password de utilizador. Para tal é necessário o envio de um objeto JSON com um endereço de email e a nova password. As respostas possíveis a todos os pedidos são:

- Status 200, significa que a requisição foi bem-sucedida.
- Status 401, indica que a solicitação não foi aplicada porque não possui credenciais de autenticação válidas.
- Status 403, expressa que o servidor compreendeu o pedido, mas não o autoriza. Semelhante ao status 401, mas neste caso, a reautorização não fará diferença. O acesso é permanentemente proibido e vinculado à lógica da aplicação, como uma password incorreta.
- Status 502, retornado pelo servidor indica que o mesmo, enquanto atua como um servidor intermediário (gateway ou proxy), recebeu uma resposta inválida do servidor para o qual a requisição foi encaminhada (upstream server).

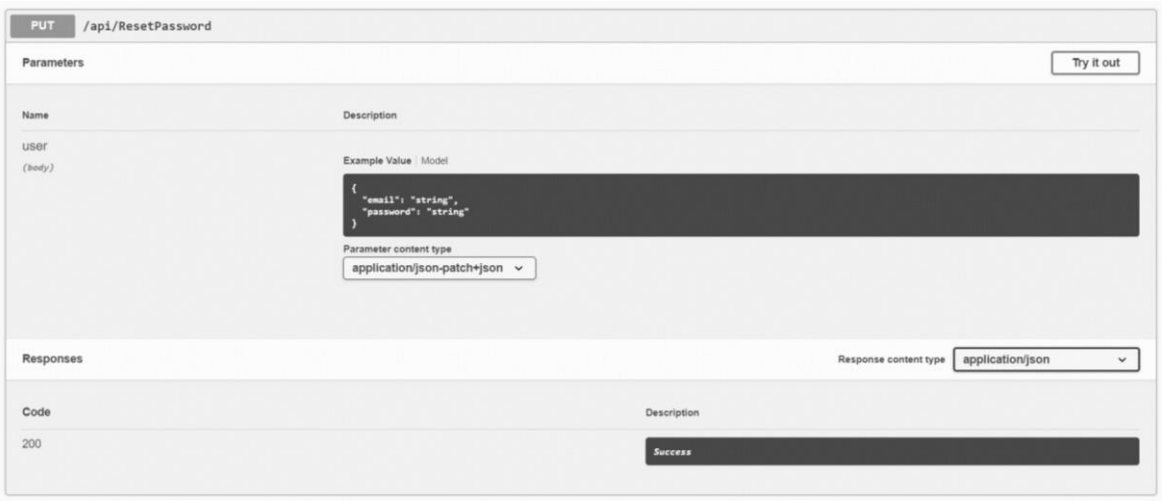


Figura 25 – Método para definir nova password.

5. RESULTADOS

5.1Testes

A tabela 6 retrata os testes realizados à Vital Band, com a respetiva duração estimada para a ocorrência desse teste e a duração realmente testada.

Tabela 6 - Tabela de testes realizados para a Vital Band.

Utilizador	Descrição		Duração Estimada	Duração Testada
1	Instalação e Configuração		2 h	6 h
	Simulação de ocorrências	Sofrer uma queda	-	1 min
		Saltar	2 min	6 min
		Dançar	10 min	20 min
		Corrida prolongada	5 min	12 min
		Várias corridas curtas	Intervalos de 30 seg	-
		Temperaturas anormais	15 min	+ de 30 min
2	Instalação e Configuração		2h	-
	Simulação de ocorrências	Sofrer uma queda	-	1 min
		Saltar	2 min	8 min
		Dançar	10 min	10 min
		Corrida prolongada	5 min	15 min
		Várias corridas curtas	Intervalos de 30 seg	-
		Temperaturas anormais	15 min	+ de 30 min
3	Instalação e configuração		2h	-
	Simulação de ocorrências	Sofrer uma queda	-	1 min
		Saltar	2 min	12 min
		Dançar	10 min	15 min
		Corrida prolongada	5 min	13 min
		Várias corridas curtas	Intervalos de 30 seg	-
		Temperaturas anormais	15 min	+ de 30 min

A tabela 7 retrata os testes realizados à RMS, com a respetiva duração estimada para a ocorrência desse teste e a duração realmente testada.

Tabela 7 - Tabela de testes realizados para a Remote Monitoring Station.

Espaço	Descrição		Duração Estimada	Duração Testada
Divisão A	Instalação e Configuração		2 h	8 h
	Aprendizagem da rotina		72 h	40 h
	Simulação de ocorrências	Televisão desligada	1 h	-
		Presença humana não detetada	1 h	8 h
		Tosse continua	5 min	5 min
		Ressonar	20 min	30 min
		Deixar cair objetos com diferentes pesos, volumes e durezas	5 seg	-
		Temperaturas anormais	15 min	+ de 30 min
		Humidades anormais	15 min	+ de 30 min
	Acionar botão SOS		-	-
	Acionar botão Helpdesk		-	-
Divisão B	Instalação e Configuração		2 h	-
	Aprendizagem da rotina		72 h	-
	Simulação de ocorrências	Televisão desligada	1 h	-
		Presença humana não detetada	1 h	8 h
		Tosse continua	5 min	5 min
		Ressonar	20 min	30 min
		Deixar cair objetos com diferentes pesos, volumes e durezas	5 seg	-
		Temperaturas anormais	15 min	+ de 30 min
		Humidades anormais	15 min	+ de 30 min
	Acionar botão SOS		-	-
	Acionar botão Helpdesk		-	-
Divisão C	Instalação e Configuração		2 h	-
	Aprendizagem da rotina		72 h	-
	Simulação de ocorrências	Televisão desligada	1 h	8 h
		Presença humana não detetada	1 h	8 h
		Tosse continua	5 min	6 min
		Ressonar	20 min	30 min
		Deixar cair objetos com diferentes pesos, volumes e durezas	5 seg	-
		Temperaturas anormais	15 min	+ de 30 min
		Humidades anormais	15 min	+ de 30 min
	Acionar botão SOS		-	-
	Acionar botão Helpdesk		-	-

A figura 26 ilustra um exemplo de sucesso, confirmado pelo código 200 (identificado a tracejado na figura 26 (a)) de um pedido efetuado à API MpDS através do Swagger. Neste exemplo em concreto, o objetivo era obter a lista de instituições às quais o utilizador tem acesso. Por motivos de segurança e sigilo a informação confidencial foi ocultada. Na figura 26 (b) o pedido não foi efetuado com sucesso, obtendo-se assim o código 500, surgindo ainda a mensagem de erro a informar que o email ou a password são inválidos.

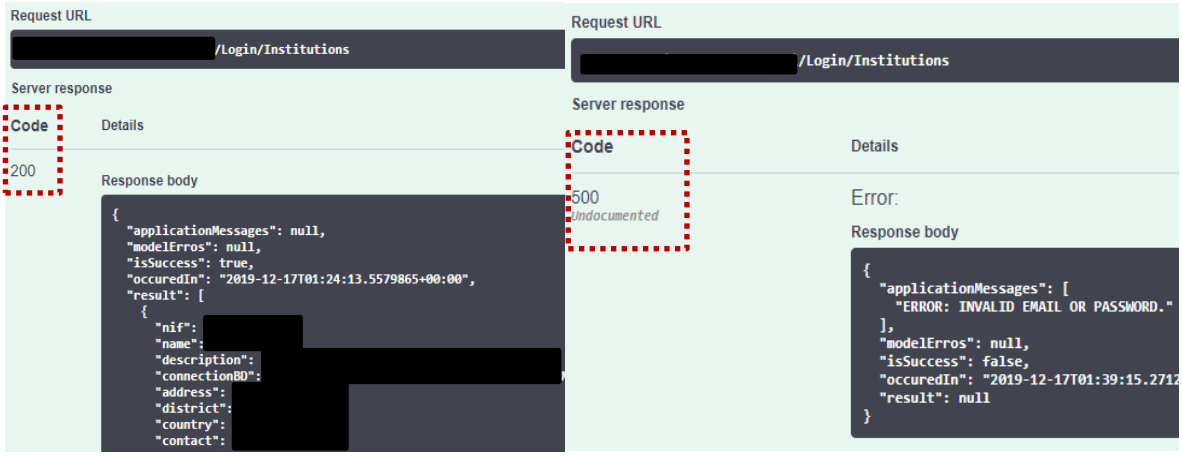


Figura 26 - Pedido para obtenção das instituições (a) com sucesso; (b) sem sucesso.

O pedido de alteração de password encontra-se representado na figura 27, sendo que, o objetivo passa por alterar a password cumprindo as normas de segurança (letras maiúsculas e minúsculas, números e caracteres especiais). Na figura 27 (a) o pedido foi efetuado com sucesso, ou seja, a password cumpria as normas de segurança e foi alterada. Sendo que posteriormente o utilizador deverá verificar o seu email para proceder à confirmação da alteração. A figura 27 (b) ilustra um pedido efetuado sem sucesso dado que a password não cumpria as normas de segurança.

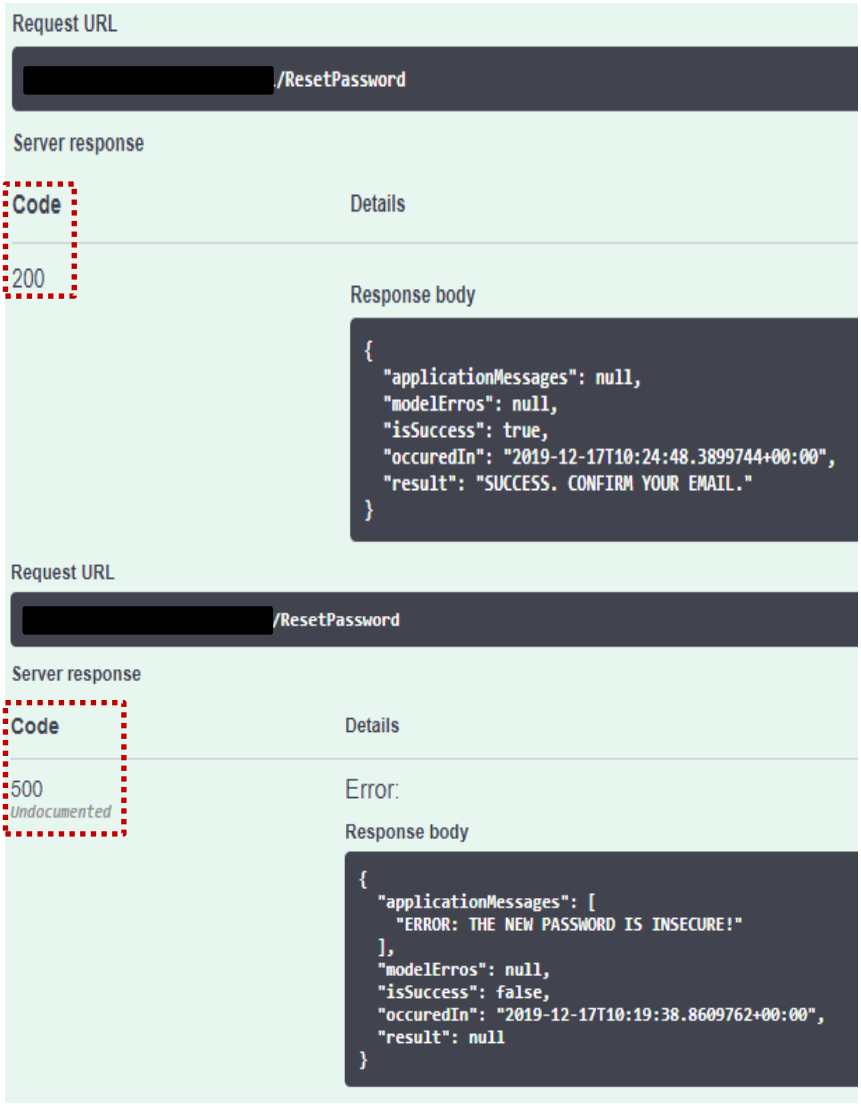


Figura 27 - Pedido de alteração de password (a) com sucesso; (b) sem sucesso.

5.2Discussão

Em relação à solução VB (Vital Band) após se iniciar sessão na plataforma Amplify é possível visualizar a localização da VB num mapa (figura 28) obtendo-se ainda informação relativamente à data e hora detetadas. De acordo com os testes efetuados, esta informação encontra-se correta. Relativamente à configuração de alertas e incidentes na aplicação Android “Gateway Software”, os mesmos sendo ativados na aplicação Android “Gateway Software” não foram apresentados na plataforma Amplify, tal como se visualiza na figura 29.

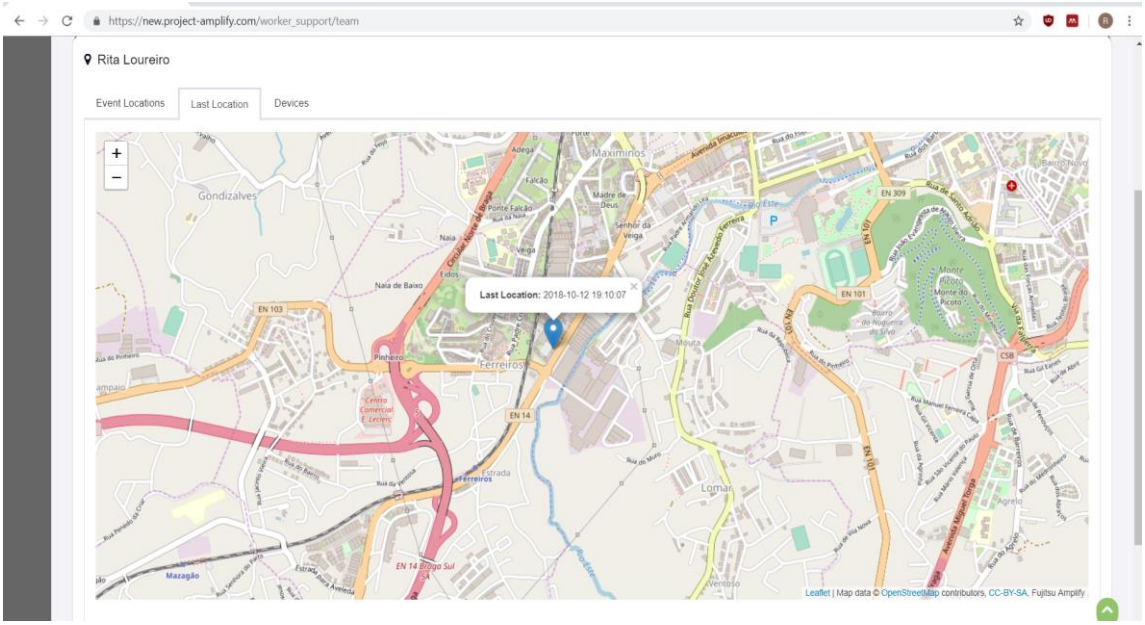


Figura 29 - Localização identificada na plataforma Amplify.

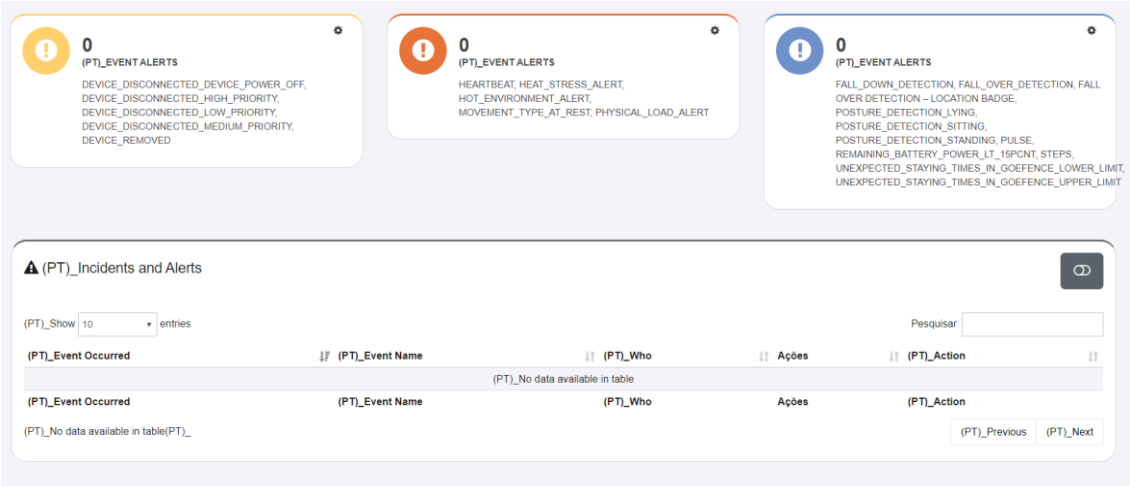


Figura 28 - Deteção de incidentes e alertas na plataforma Amplify.

Apesar da unidade de sensor efetuar vibrações, por parte do utilizador as mesmas não eram compreendidas nem associadas a um alerta concreto. Também as variáveis de ambiente, como a temperatura, humidade, etc, não foram apresentadas na plataforma Amplify (figura 30).

Para a solução RMS foi testado um servidor SIP, o MizuDroid, dado que o servidor SIP aconselhado pela Fujitsu não foi possível de instalar. Numa primeira fase testou-se entre dois smartphones Android, tendo sido efetuada a comunicação com sucesso. Contudo a dificuldade de estabelecer um servidor SIP para chamadas de emergência, não permitiu que se obtivessem resultados em relação a esta solução.

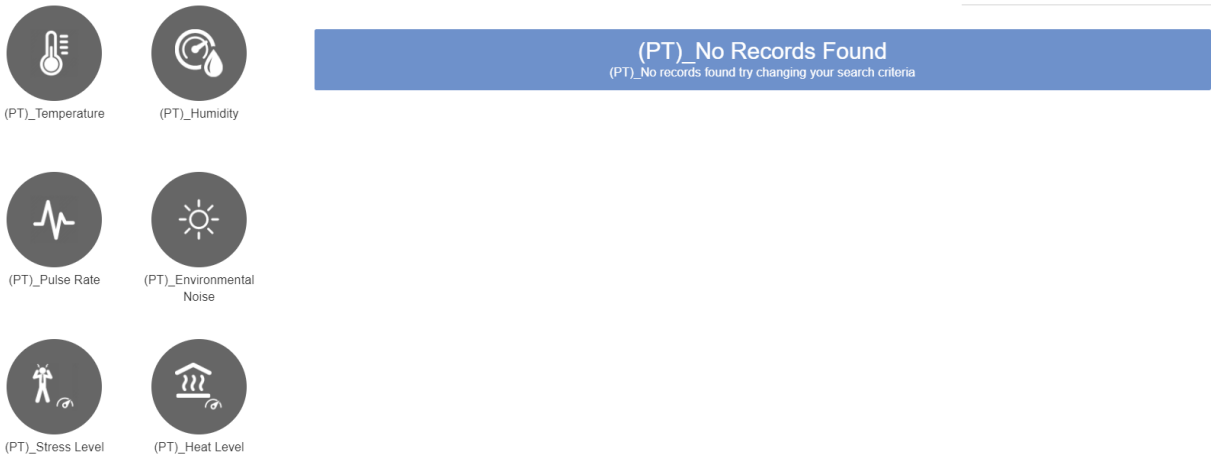


Figura 30 - Variáveis de ambiente disponíveis na plataforma Amplify.

Esta dificuldade em grande parte deveu-se ao facto de este ser um produto descontinuado em Portugal, e os recursos especializados nesta solução se encontrarem apenas no Japão.

Em relação ao projeto MpDS, o Swagger UI permitiu numa primeira fase testar de forma rápida e eficiente todos os pedidos que deveriam ser efetuados pela aplicação MpDS, detetando-se assim de forma precoce problemas que poderiam ocorrer em clientes.

Este projeto, apesar de ainda não ser uma total solução IoT apresenta uma interação com diferentes softwares de gestão de cuidados de saúde, assim como aplicações móveis e gestor de conteúdo. Apesar disso, é um projeto que em termos futuros e após a análise desta dissertação poderá existir uma forte aposta de modo a ter-se uma solução IoT.

Concluídos os testes de simulação das três soluções abordadas, cujo principal objetivo era identificar o nível de maturidade das soluções e o desempenho em ambiente de simulação, é possível identificar os resultados obtidos.

Um dos aspetos importantes, obtidos nos testes é o facto da aplicação MpDS possuir proteção integrada contra perda de sinal, ou seja, caso isso ocorra, o utilizador é alertado e os eventos são armazenados em cache e transferidos, posteriormente, para a cloud quando o sinal é restaurado.

Os resultados obtidos apontam para uma imaturidade das soluções VB e RMS, apesar dos conceitos e os protocolos estarem consolidados, verificou-se que nem sempre a solução funcionava da forma que a documentação descrevia, sendo difícil testar em ambiente de simulação, como foi o caso dos testes desta dissertação, ou seja, por parte da VB nunca foi compreendidos pelo utilizador a que tipo de eventos correspondiam as vibrações sentidas.

6. CONCLUSÃO

A presente dissertação é concluída neste capítulo. Ao longo deste, é realizada uma reflexão sobre os objetivos propostos e expostas as contribuições da dissertação. São ainda apresentadas direções para trabalho futuro.

6.1Conclusões

O trabalho elaborado ao longo desta dissertação, tinha como finalidade estudar o conceito da Internet das Coisas. Para tal foram estabelecidos objetivos mais específicos, tais como, identificação dos princípios e paradigmas associados à Internet das Coisas, investigação e análise de arquiteturas de referência e de modelos de referência, estudo e análise das plataformas de *middleware* para IoT*,* levantamento dos protocolos, plataformas IoT e desafios. Estabelecendo posteriormente uma relação entre as soluções estudadas, os protocolos e as arquiteturas de referência.

Considerando os objetivos inicialmente propostos conclui-se que:

- O conceito de Internet das Coisas é de fácil compreensão, contudo os desafios e os custos de implementação deste conceito num projeto, requerem um estudo aprofundado das arquiteturas de referências a adotar, dado que estas têm vindo a aumentar e são um fator importante na implementação de um projeto.
- Existe uma forte adesão por parte de empresas e universidades, como é o caso da Cisco, Dell, Intel, Microsoft, ARM e Universidade de Princeton, que se uniram e criaram o consorcio OpenFog. O objetivo destes consórcios é criar documentação, regulamentação e arquiteturas especificas para IoT, de modo a colmatarem as dificuldades existentes em projetos IoT.
- A análise da arquitetura de referência IoT-A permitiu obter uma visão funcional da mesma, tendo sido abordados os vários elementos que a compõem. Obtendo-se assim uma base sólida para o desenvolvimento de um projeto IoT.
- O projeto OpenIot representa um enorme destaque em termos de plataformas de *middleware* para IoT, tal se deve à consolidação da informação e à dedicação e renome dos autores.

- As soluções da Fujitsu selecionadas para teste, nomeadamente a VB e a RMS, requerem uma maior aposta na melhoria da documentação, nas interfaces das aplicações Android assim como na interface da plataforma Amplify. Apesar destes dois produtos apresentarem um conceito interessante e viável para o mercado das soluções IoT, em termos de simulação de testes para esta dissertação relevaram-se produtos imaturos, de difícil acesso e compreensão da logística de simulação. O facto de a solução RMS ser um produto que desde o início do ano de 2019 deixou de ser comercializado na Europa, sendo o Japão o seu mercado alvo, dificultou a aprendizagem desta solução dado que o staff especializado se encontra no Japão.
- O projeto MpDS com o auxílio do Swagger UI, relevou ter uma API de fácil integração com diferentes sistemas, tais como aplicação Android MpDS, aplicação multiplataforma Second Opinion MpDS e software de gestão de cuidados de saúde, nomeadamente o software WinGCS.

A principal contribuição desta dissertação foi o facto de se materializarem os conceitos associados à Internet das Coisas, analisarem arquiteturas de referência e plataformas de *middleware* relevantes para um projeto IoT, assim como o estudo, análise e comparação de soluções de mercado com os conceitos, protocolos IoT e arquiteturas abordadas.

6.2Trabalho Futuro

Neste tipo de investigação académica, por vezes as situações de simulação revelam-se um problema dado que a documentação requeria mais informação.

Relativamente à Vital Band, o facto de a mesma possuir um amostrador para as horas e para a data seria um ótimo upgrade para quem utiliza esta pulseira diariamente no seu trabalho. Assim como um amostrador com mensagens ou ícones de forma a que o utilizador compreendesse melhor os alertas. O trabalho realizado deixa em aberto algumas questões sobre a maturidade das soluções IoT da Fujitsu.

Em relação ao sistema MpDS seria vantajoso uma versão da aplicação para relógios *wearables*, com possibilidade de pesquisa de um paciente na base de dados, abertura de um novo caso clínico, visualização detalhada da informação relevante do paciente, etc. Uma integração possível seriam as pulseiras/relógios de fitness, tais como, Xiaomi Mi Band,

Samsung Galaxy Fit, Garmin Vivosmart, etc, de modo a obter-se a qualidade de sono, os níveis de exercício físico diário, o ritmo cardíaco. Através destes dados seria possível o profissional de saúde analisar o nível de sedentarismo do paciente, assim como a qualidade e o número de horas de sono, podendo posteriormente aconselhar o paciente a ter hábitos de vida mais saudáveis.

REFERÊNCIAS

3cx. (19 de Abril de 2019). *Dúvidas sobre SDP: O que é Session Description Protocol e seus usos*. Obtido de 3cx: https://www.3cx.com.br/voip-sip/sdp/

3cx. (19 de Abril de 2019). *O que é um servidor SIP?* Obtido de 3cx: https://www.3cx.com.br/voip-sip/sip-server/

Agrawal, R. (9 de Maio de 2010). *Messaging Using AMQP*. Obtido de SlideShare: https://www.slideshare.net/rahula24/amqp-basic

Belfiore, M. (2014). How to Benefit From the Internet of Things. *RFID Journal*.

Callcentermagazine. (30 de Junho de 2015). *Em 2020, haverá 50 mil milhões de dispositivos conectados à Internet*. Obtido de Callcentermagazine: https://www.callcentermagazine.net/tecnologia/em-2020-havera-50-mil-milhoes-de-dispositivos-conectados-a-internet/

Cetax. (Julho de 2017). *APACHE HADOOP: O QUE É, CONCEITO E DEFINIÇÃO*. Obtido de Cetax: https://www.cetax.com.br/blog/apache-hadoop/

CyberVision. (14 de Outubro de 2018). Obtido de Kaa Project: http://kaaproject.github.io/kaa/docs/v0.10.0/Welcome/

HiveMQ. (2015). *MQTT 101*. Obtido de HiveMQ: https://www.hivemq.com/blog/how-to-get-started-with-mqtt

Lee, J. (2018 de Outubro de 14). *How to Choose the Right IoT Platform: The Ultimate Checklist*. Obtido de Hackernoon: https://hackernoon.com/how-to-choose-the-right-iot-platform-the-ultimate-checklist-47b5575d4e20

ManSystems. (24 de Agosto de 2015). *10 Impressive facts about Internet of Things*. Obtido em 2018, de https://www.mansystems.com/10-impressive-facts-about-internet-of-things/

OMG (Object Management Group). (8 de Novembro de 2019). *DDS – The Proven Data Connectivity Standard for IoT.* Obtido de twinoakscomputing: http://twinoakscomputing.com/datasheets/DDS-Brochure.pdf

Paulo F. Pires, F. C. (2015). Plataformas para a Internet das Coisas. Em F. C. Paulo F. Pires, *Minicursos do XXXIII Simpósio Brasileiro de Redes de Computadores e Sistemas*

Distribuídos (pp. 110-169). Porto Alegre, RS, Brasil. Obtido de http://www.dimap.ufrn.br.

Rajkumar Buyya, A. V. (2016). *Internet of Things Principles and Paradigms.* Morgan Kaufmann .

RF Wireless World. (s.d.). *DDS Protocol Architecture basics | DDS Protocol in IoT*. Obtido de rfwirelessworld: https://www.rfwireless-world.com/Terminology/DDS-protocol-architecture.html

ANEXO I. VITAL BAND (VB)

Tabela 8 – Sintetização da informação dos estados da unidade de sensor da Vital Band.

Estado	Ação	Displays	Descrição
Ligado	Mantenha o botão de energia pressionado entre 3 a 5 segundos	Pisca verde duas vezes	Ligado
		Pisca verde duas vezes, depois pisca vermelho uma vez	Se a bateria estiver muito fraca, o dispositivo irá desligar-se automaticamente.
Desligado	Mantenha o botão de energia pressionado entre 5 a 9 segundos	Pisca verde três vezes	Desligado
Verificação da bateria	Mantenha o botão de energia pressionado por 1 segundo	Pisca verde três vezes	A bateria tem energia suficiente
		Pisca vermelho três vezes	A bateria está fraca (menos de 20%)
Carregamento	_____	Permanece vermelho	Permanece acesa durante o carregamento. O LED apaga quando o carregamento estiver completo
Erro no carregamento		Pisca vermelho	Há um erro de carregamento ou falha no hardware
Alerta de deteção	_____	15 longos flashes vermelhos	Alerta de deteção de queda
Configuração de update	_____	Permanece laranja	A configuração da Vital Sensing Band está a ser atualizada

ANEXO II. REMOTE MONITORING STATION (RMS)

Tabela 9 - Lista de funções da Remote Monitoring Station (RMS).

Funções	Especificações
Deteção de som	Deteta conversas e fala
Deteção de ronco	Deteta a frequência / intervalo de ronco
Deteção de distúrbios respiratórios	Deteta longos intervalos de silêncio entre os roncos.
Deteção de tosse	Deteta alta frequência de tosses consecutivas. Detetado como positivo se pelo menos 10 séries de tosses consecutivas (ou seja, 20 tosses) são detetadas em uma hora.
Deteção de som de televisão	Deteção de som proveniente da televisão
Medição de pressão sonora	Mede a pressão sonora por minuto
Deteção de ruido alto anormal	Emite notificações de pressão sonora anormalmente alta.
Medição da temperatura e da humidade	Medição da temperatura/humidade
Estimativa do nível do ambiente térmico	Estimativas do nível do ambiente térmico em uma escala de 1 a 5 através do sensor termo-higrómetro.
Notificação de temperatura/risco de insolação	Se o ambiente térmico estiver com valores elevados, o administrador é notificado e o terminal reproduz automaticamente um anúncio com as medidas devem ser tomadas.
Deteção de movimento humano	Utiliza o sensor de movimento para detetar o movimento humano, de modo a determinar se existe alguém em seu redor.
Camara	Pode fotografar imagens estáticas
Chamada VoIP originada	Notifica/Alerta o administrador simplesmente pressionando o botão Chamada de Emergência ou Consulta.
Chamada VoIP recebida	Recebe chamadas VoIP efetuadas pelos administradores.
Transmissão de notificação	Envia uma notificação de que o botão de chamada de emergência ou o botão de chamada de consulta foi pressionado.
Cancelar	Pressionar o botão “Cancelar” chamada cancela a notificação de deteção de anormalidade. Se o botão de chamada de emergência ou o botão de chamada de consulta for pressionado acidentalmente, ou se uma anomalia for detetada erroneamente, o utilizador poderá impedir que a chamada seja conectada ou que uma foto seja tirada.

Reprodução de som em simultâneo	Cinco arquivos de áudio são armazenados no terminal como meio de comunicação com o utilizador. É possível atualizar arquivos de áudio pela rede.
Modo de prevenção transgredido	Liga/Desliga o modo de prevenção contra transgressão. Quando o sensor de som ou de movimento é detetado na ausência do utilizador, o servidor é notificado do evento de invasão, emite um bipe de alarme, tira uma foto e faz o upload para o servidor.
Modo de agendamento preventivo ultrapassado	Permite que os utilizadores designem a data e a hora para ativar/desativar o modo de prevenção de transgressão. Automaticamente ativa o modo de prevenção contra invasão quando a data e a hora designadas são atingidas. * Por favor, desligue o modo de prevenção contra transgressão antes de ligar este modo.
Update de firmware da RMS	Faz o download do firmware do RMS do servidor e o atualiza-o.

Tabela 10 - Lista informativa sobre a luz LED da Remote Monitoring Station (RMS).

LED	**Estado da Unidade Principal**	**Estado do LED**
Estado LED (Verde ou Vermelho)	Chamada	A piscar verde
	Anúncio de emergência	
	Anúncio de distúrbio de calor perigoso	
	Conversar	Luz verde
	Anormalidade do equipamento em progresso	A piscar vermelho
	Update de *firmware* em progresso	A piscar rapidamente verde
LED Ligado/Desligado (verde)	Ligado	Ligado
	Desligado	Desligado
Camara LED (verde)	Camara utilizada	Ligado (20 segundos depois de desligar)
	Camara não utilizada	Desligado
LED de conexão WLAN (verde)	Configuração WLAN em progresso	A piscar verde
	Configuração WLAN completa	Luz verde (30 segundos depois de desligar)
	Possibilidade de comunicação com a rede	Luz verde

LED de emparelhamento Bluetooth	Emparelhamento em progresso	A piscar verde (até 30 segundos)
	Emparelhamento estabelecido	Luz verde (3 segundos depois de desligar)
	Emparelhamento falhado	A piscar vermelho (3 segundos depois de desligar)